

Partie 4: Détection des contours, Segmentation

Pierre Maurel

Visages, IRISA/INRIA

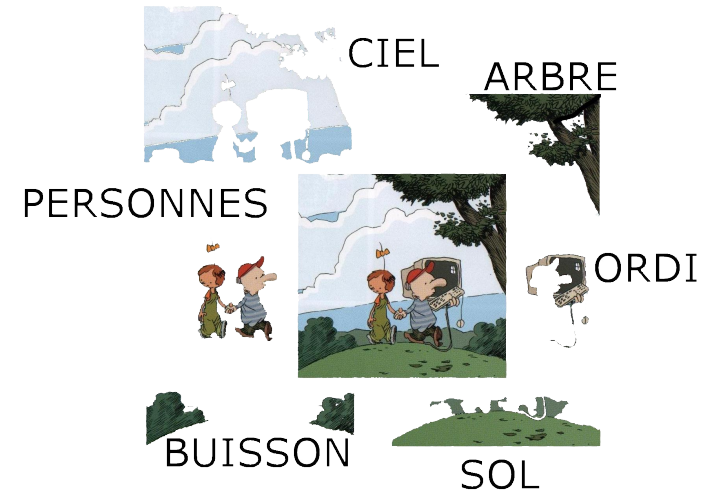
pierre.maurel@irisa.fr

<http://www.normalesup.org/~pmaurel/IMA/>

1/149

Analyse d'images, vision par ordinateur

- Segmentation : partitionner l'image en ses différentes parties.
- Reconnaissance : étiqueter les différentes parties

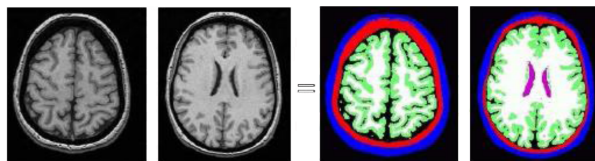


2/149

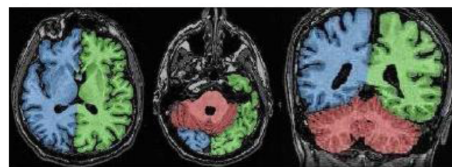
Segmentation ?

- La segmentation vise à découper une image en régions connexes présentant une homogénéité selon un certain critère.
- différentes possibilités → fonction de ce qu'on veut en faire
- Exemples

Peau, os, LCR, matière grise, matière blanche, ventricules



Hémisphère gauche, hémisphère droit, cervelet



Images issues de l'HDR de J.F. Mangin

3/149

Segmentation ?

À quoi ça sert ?

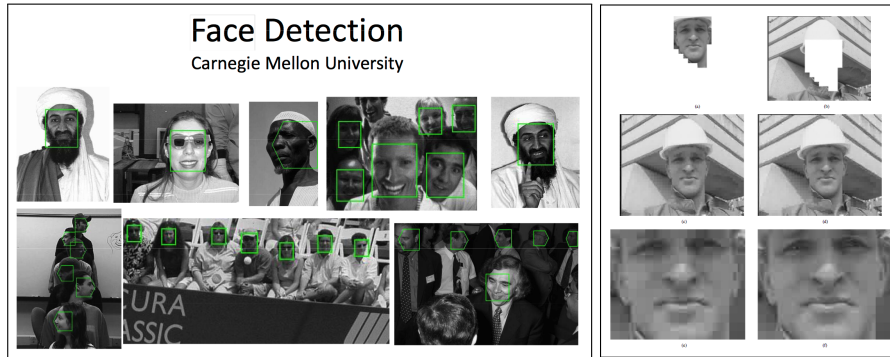
Important prérequis pour les étapes de mesure, de compréhension de la scène :

- reconnaissance d'objets
- indexation : rechercher dans une base d'images, les images "ressemblantes" à une image initiale
- compression
- recalage d'images, mises en correspondance
- ...

4/149

Segmentation ?

Exemple d'applications : Segmentation de visages

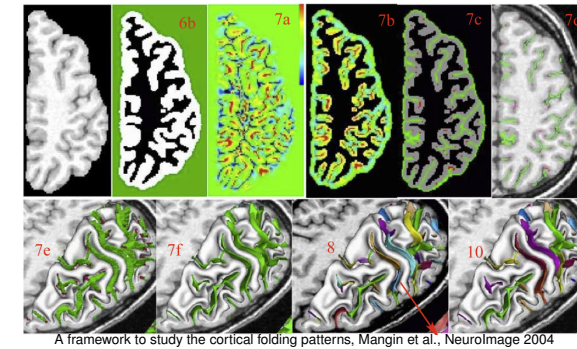


- reconnaissance
- compression
- mise au point automatique

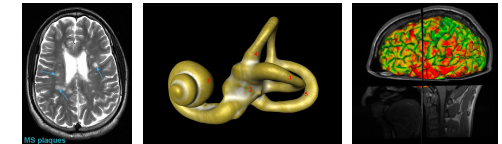
5 / 149

Segmentation ?

Exemple d'applications : Imagerie Médicale



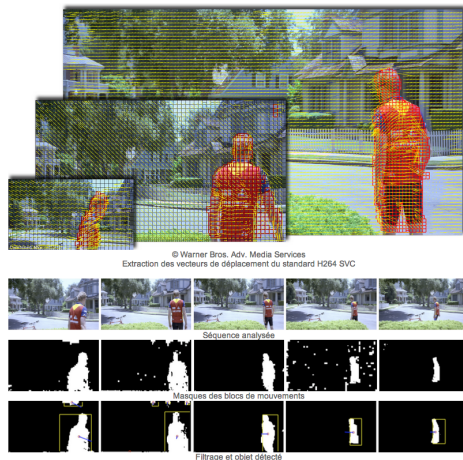
- Quantification des volumes des tissus, des organes
- Localisation d'une pathologie
- Étude d'une structure anatomique
- Planification d'un traitement
- Chirurgie assistée par ordinateur



6 / 149

Segmentation ?

Exemple d'applications : segmentation de vidéos



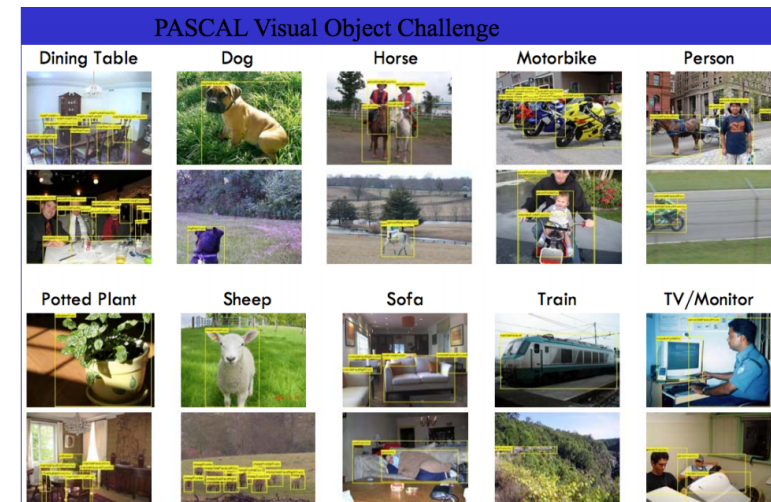
- Suivi d'objets/personnes
- Compression
- Reconnaissance

<http://www.labri.fr/projet/AIV/segmentationindexation.php>

7 / 149

Segmentation ?

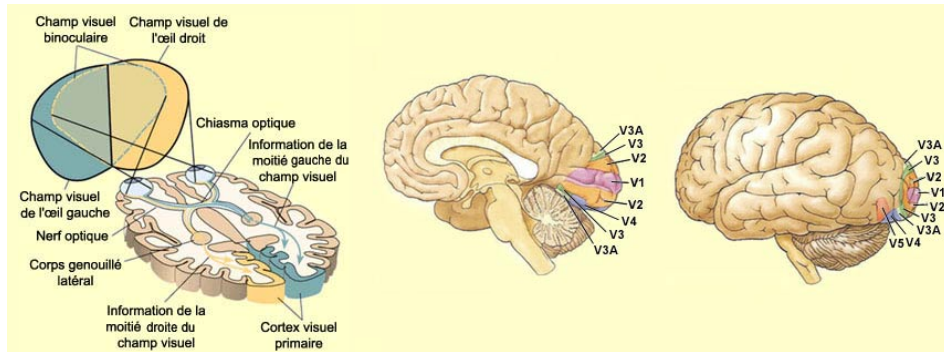
Exemple d'applications : reconnaissance d'objets



8 / 149

Segmentation ?

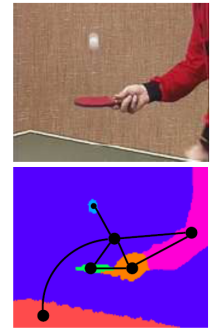
- Dans le système visuel, on a montré que les aires V1 et V2 sont sensibles à l'orientation du stimulus et que V3 et V4 extraient des contours.



9 / 149

Segmentation ?

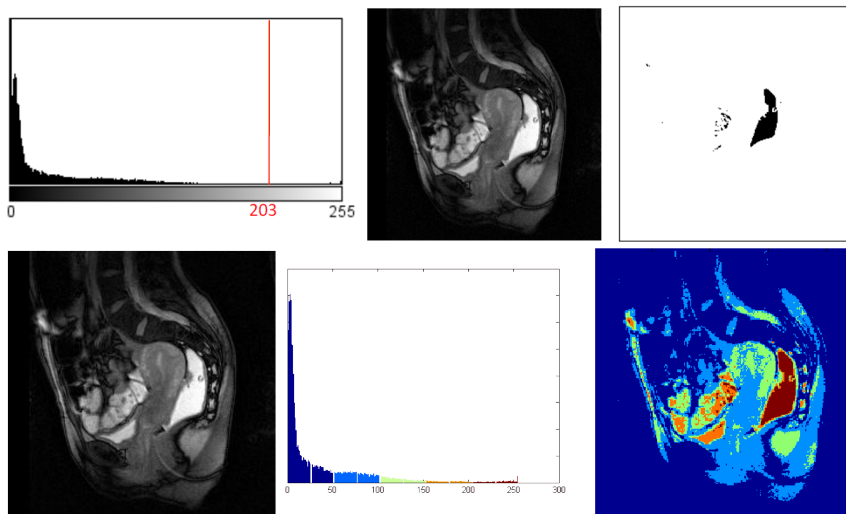
- Description "haut-niveau" d'une image
- Représentation sous forme de graphe d'adjacence
- La segmentation peut être basée sur
 - les discontinuités de l'image (contours)
 - les similitudes entre région (couleur, intensité, texture ...)
- Pas de solution universelle : en général, algo limité à un type d'application et/ou d'image
- Différentes approches :
 - approches globales
 - approches régions
 - approches contours



10 / 149

Approches Globales

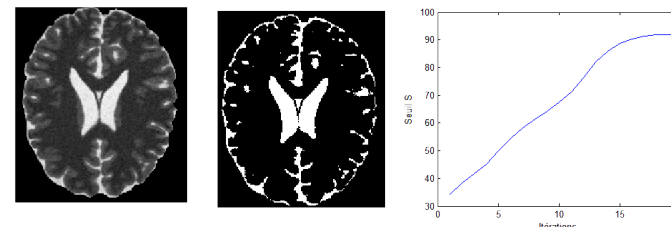
exemple le plus simple : seuillage d'histogramme



12 / 149

Seuillage d'histogramme

- Détermination du (ou des) seuil(s)
 - valeur obtenue par test
 - valeur moyenne
 - valeur médiane
 - choix automatique
- Un algorithme simple
 - Choisir un seuil S initial (moyenne, médiane, ...)
 - On seuille \rightarrow 2 groupes de pixels de moyenne μ_1 et μ_2
 - On calcule $S = \frac{\mu_1 + \mu_2}{2}$
 - On itère jusqu'à ce que S soit constant



DEMO MATLAB

13 / 149

Seuillage d'histogramme

Méthode d'Otsu (1979)

- Un seuil t définit deux groupes de pixel : C_1 et C_2
- On cherche alors le seuil qui minimise la variance intra-classe :

$$\sigma_w^2(t) = \omega_1(t)\sigma_1^2(t) + \omega_2(t)\sigma_2^2(t)$$

- Les poids $\omega_i(t)$ représentent la probabilité d'être dans la i ème classe
- les σ_i^2 sont les variances de ces classes



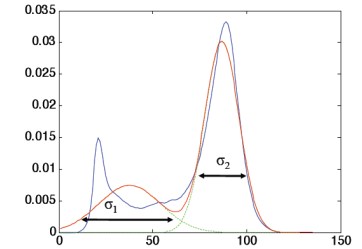
<http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html>

Seuillage d'histogramme

Seuillage par classification bayésienne

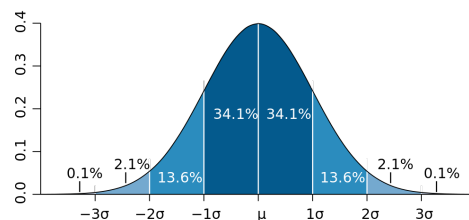
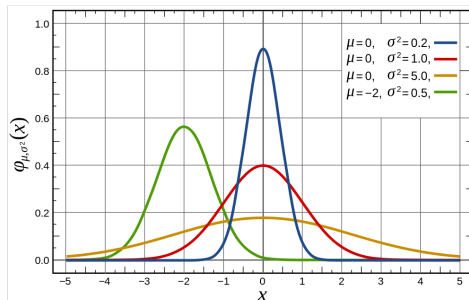
- Approximation de l'histogramme par un mélange de gaussiennes

$$\begin{cases} p_1(x) = \frac{P_1}{\sigma_1\sqrt{2\pi}} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} \\ p_2(x) = \frac{P_2}{\sigma_2\sqrt{2\pi}} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}} \end{cases} \quad \text{et} \quad P_1 + P_2 = 1$$

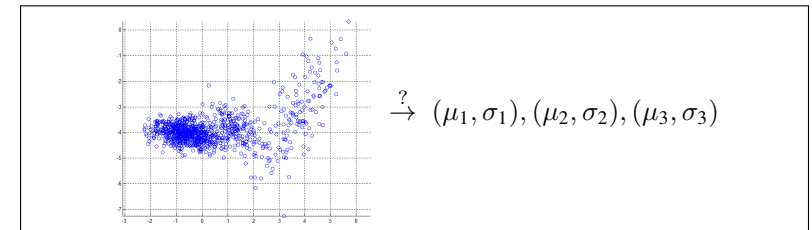


- Estimation de 5 paramètres libres (EM, gradient)

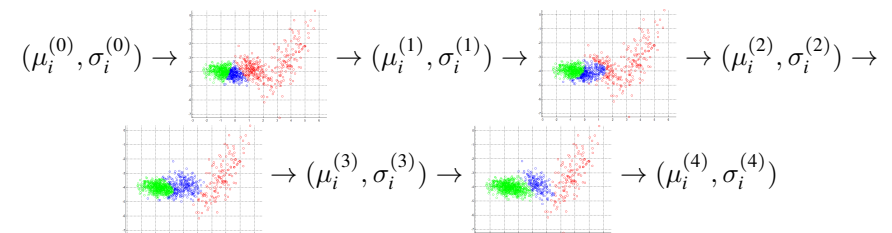
Aparté : Loi gaussienne



Algorithme EM, Modèle de mélanges gaussiens



DEMO MATLAB



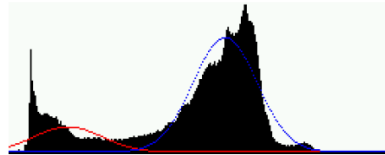
Seuillage d'histogramme

Seuillage par classification bayésienne



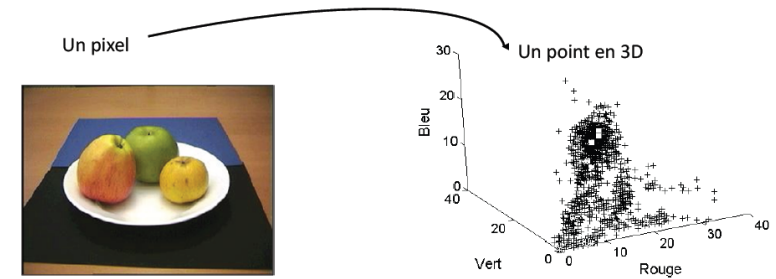
Image originale

Image segmentée



Algorithme des K-moyennes (K-means)

- Algorithme de classification dans un espace n -dimensionnel
- ici : $n = 1$ (image en niveaux de gris) ou $n = 3$ (image en couleurs) ou plus



Algorithme des K-moyennes (K-means)

- On initialise K graines (aléatoires par ex.) étiquetées de 1 à K
- On répète, jusqu'à convergence :
 - 1) Pour chaque pixel, on trouve la graine i la plus proche au sens de la distance euclidienne
 - 2) On donne à ce pixel l'étiquette de la graine i
 - 3) On calcule le barycentre de chaque classe → les barycentres deviennent les nouvelles graines



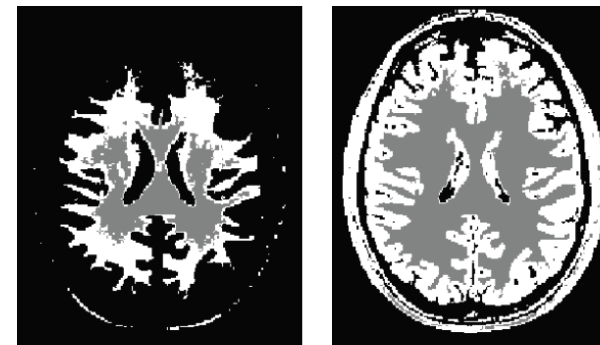
1) k initial "means" (in this case $k=3$) are randomly selected from the data set (shown in color).
2) k clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.
3) The centroid of each of the k clusters becomes the new means.
4) Steps 2 and 3 are repeated until convergence has been reached.

WIKIPEDIA

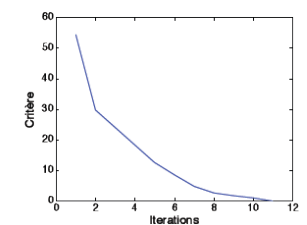
Algorithme des K-moyennes (K-means)

- Algorithme des K-moyennes en 1D

Initialisation (103,239,234) Segmentation finale (19,219,114)



Convergence du critère



Algorithme des K-moyennes (K-means)

Inconvénients

- Sensibilité à l'initialisation
- Choix du nombre de classe k



22 / 149

variante : Algorithme des Fuzzy c-means

- Fuzzy C-Means : chaque point a un degré "flou" d'appartenance à chaque classe
- On donne maintenant un poids d'appartenance d'un pixel s à une classe $k : u_{sk}$ tel que $\sum_k u_{sk} = 1, \forall s$
- ainsi, la moyenne de la classe k devient :

$$\mu_k = \frac{\sum_s u_{sk} I(s)}{\sum_s u_{sk}}$$

- le problème devient donc maintenant : trouver μ_1, \dots, μ_K et $U = (u_{sk})$ tels que

$$\sum_k \sum_s u_{sk}^m |I(s) - \mu_k|^2 \text{ soit minimal.}$$

$m > 1$ est un paramètre constant (degré de flou / fuzziness)

23 / 149

variante : Algorithme des Fuzzy c-means

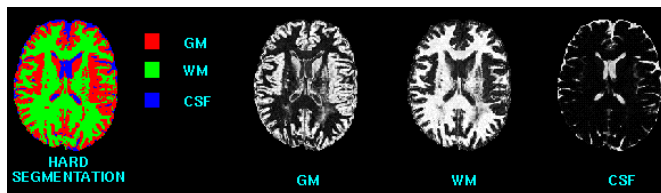
- Ici encore : algorithme itératif

1 Répéter

$$1 \quad u_{sk} = \sum_{l=1}^K \left(\frac{|I(s) - \mu_k|}{|I(s) - \mu_l|} \right)^{-\frac{2}{m-1}}$$

$$2 \quad \mu_k = \frac{\sum_s u_{sk} I(s)}{\sum_s u_{sk}}$$

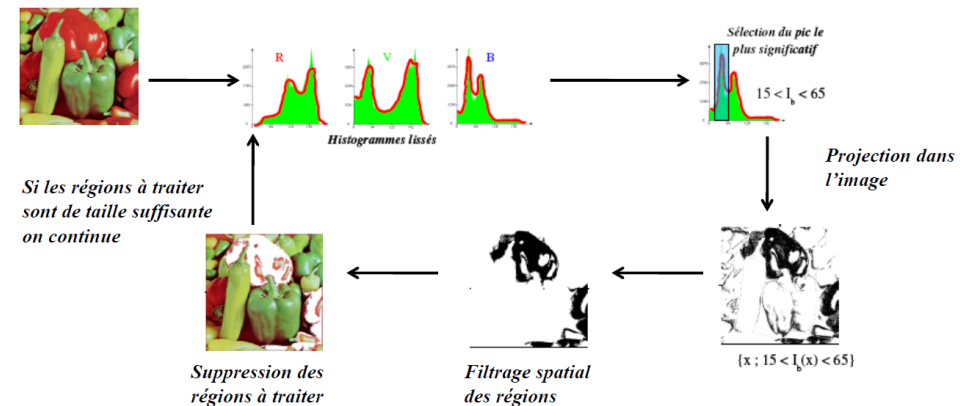
- 2 jusqu'à ce que $\max(|u_{sk}^n - u_{sk}^{n-1}|) < \epsilon$



24 / 149

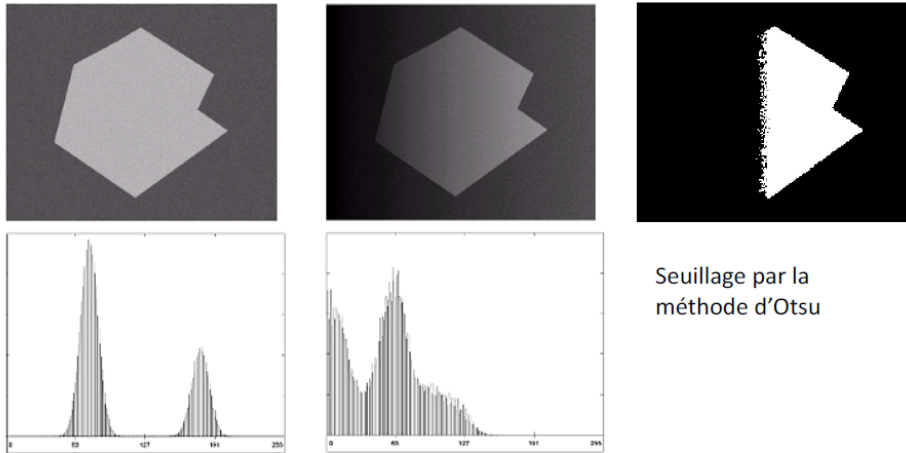
Sélection récursive d'histogramme

Ohlander, Price et Reddy (1978)



25 / 149

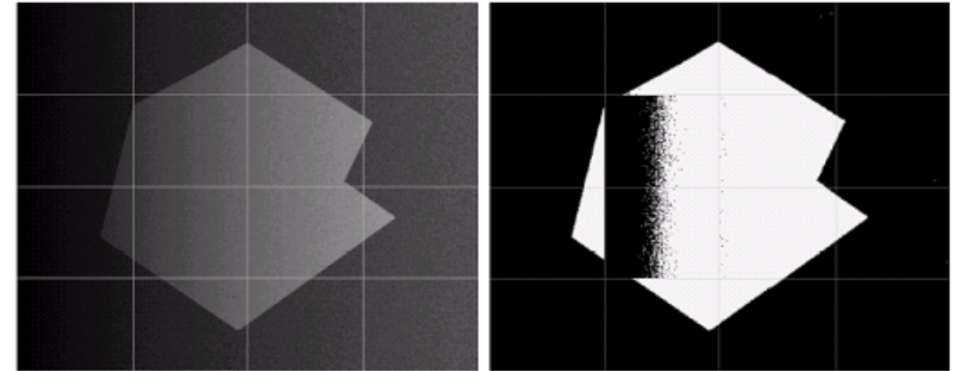
Limite des approches globales



26 / 149

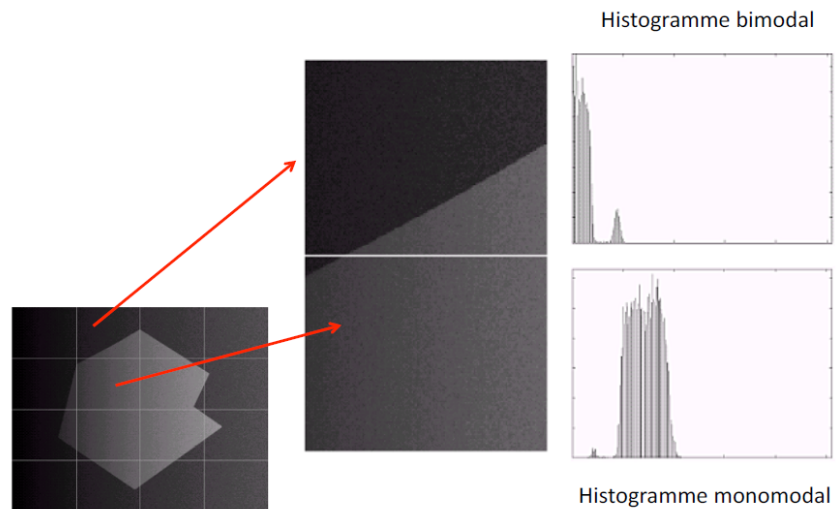
Seuillage adaptatif

- On divise l'image en un certain nombre de sous-régions
- → seuillage sur chaque région



27 / 149

Seuillage adaptatif

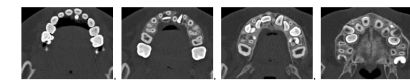


28 / 149

Approches globales : bilan

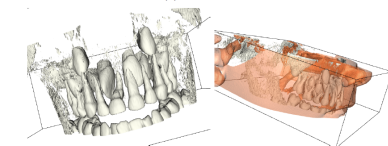
- **Avantages :**
 - simple, rapide
 - bien adapté aux histogrammes multimodaux
- **Inconvénients :**
 - il faut connaître le nombre de classes
 - choisir les seuils
 - Pas d'information de connexité

En CT-X ...



(a) 4 plans de coupe

Parfois suffisant :



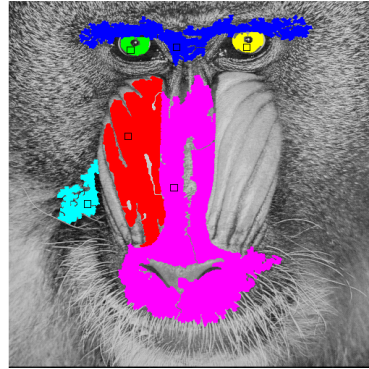
(b) Seuillage à 80

(c) Seuillage à 500

29 / 149

Croissance de régions - Region growing

- on choisit une (ou plusieurs) “graine(s)”
- La région R (pour l’instant réduite à un point) possède une moyenne μ_R et un écart-type σ_R
- On ajoute alors à R tous les pixels voisins de R qui sont suffisamment semblables à R , exple



$$|I(x) - \mu_R| < \text{seuil}$$

ou bien

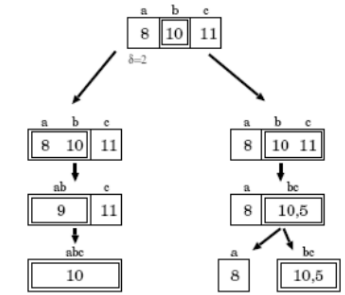
$$\begin{cases} \min\{|I(x) - I(y)|; y \in R \cap V(x)\} < \text{seuil} \\ |I(x) - \mu_R| < \sigma_R \end{cases}$$

31 / 149

Croissance de régions - Region growing

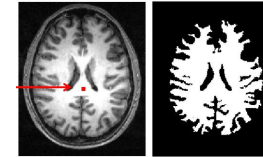
Limitations :

- Influence du choix des graines
- Influence de l’ordre de parcours des points de la frontière
- choix du seuil



Avantage :

- Implémentation : très rapide, si l’on utilise une structure de données adaptée (files d’attente).



32 / 149

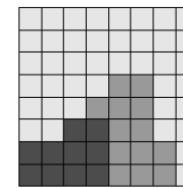
Split and Merge

- Algorithme “split and merge” [Pavlidiset Horowitz, 1974]
 - Le principe consiste à (sur-)diviser l’image en régions homogènes (split) que l’on va ensuite regrouper (merge)
 - étape **split** : on crée une partition de l’image par division récursive en régions de taille identique lorsqu’un critère d’homogénéité n’est pas satisfait.
 - étape **merge** : on utilise le graphe d’adjacence créé pendant le **split** pour regrouper des régions voisines et suffisamment homogènes.

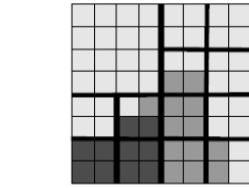
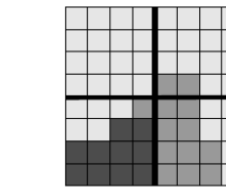
33 / 149

Split and Merge

- Illustration de l’algorithme : **SPLIT**



Quadtree



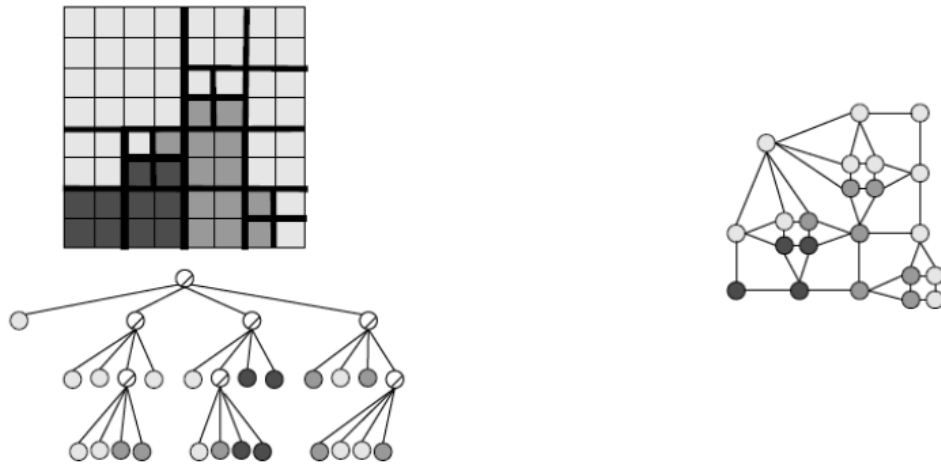
Graphe d’adjacence



34 / 149

Split and Merge

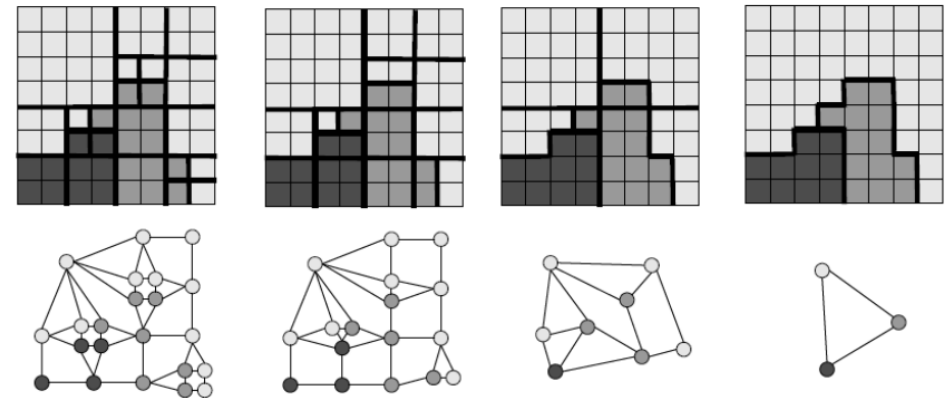
- Illustration de l'algorithme : **SPLIT**
 - Résultat final de cette étape : sur-segmentation



35 / 149

Split and Merge

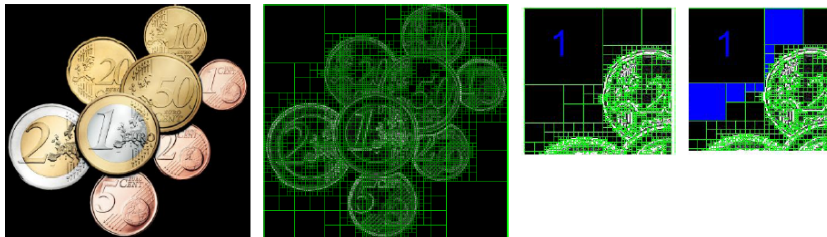
- Illustration de l'algorithme : **MERGE**



36 / 149

Split and Merge

- La phase SPLIT crée une sur-segmentation de l'image que la phase MERGE vient corriger
- La phase MERGE → croissance de régions



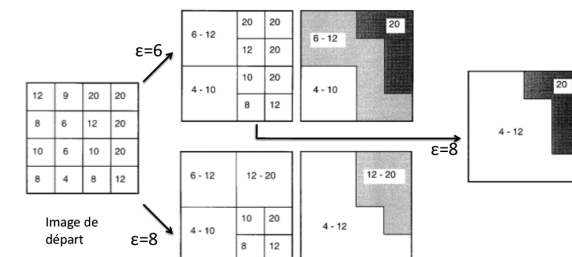
- critère d'homogénéité :
 - sur les extrema des régions $\max_R(I(x, y)) - \min_R(I(x, y)) < \epsilon$
 - sur la variance au sein de la région $\sum_R (I(x, y) - \mu_R)^2 < \epsilon$

37 / 149

Split and Merge

Amélioration

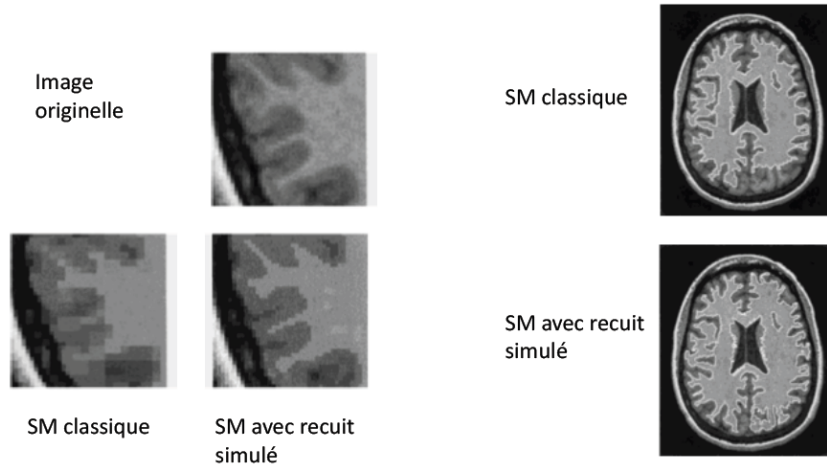
- une sorte de recuit simulé (Manousakas et al. 1998)
- constat :
 - ϵ trop grand → splits pas assez nombreux
 - ϵ trop petit → pas assez de regroupements
- On utilise $\epsilon/2$ pour la phase de split puis on regroupe en passant progressivement de $\epsilon/2$ à ϵ



38 / 149

Split and Merge

Amélioration



39 / 149

Split and Merge

Avantage :

- méthode hybride (locale/globale)

Inconvénients :

- crée des structures carrées dans la segmentation de l'image
- comme pour la croissance de régions : sensibilité à l'ordre de parcours des régions

Il existe encore d'autres méthodes

- CSC (Color Structure Code), Rehrmann et Prieese (1993)
- MDL (Minimum Length Description)

40 / 149

Méthodes markoviennes

- En restauration d'images (méthodes variationnelles), on cherche à minimiser une énergie de la forme : $E_{\text{données}}(u) + E_{\text{régularité}}(u)$
- Dans le cadre bayésien, on donne une interprétation statistique
- Formule de Bayes :



$$\mathbf{P}(A|B) = \frac{\mathbf{P}(B|A) \mathbf{P}(A)}{\mathbf{P}(B)}$$

- On considère donc nos images comme des réalisations de variables aléatoires : $\mathbf{P}(u|f) = \frac{\mathbf{P}(f|u) \mathbf{P}(u)}{\mathbf{P}(f)}$ où f est l'image observée (à traiter), et u est l'inconnue.



41 / 149

Méthodes markoviennes

- On définit/modélise donc $\mathbf{P}(f|u)$ et $\mathbf{P}(u)$
- Puis on cherche le u qui **maximise** la probabilité $\mathbf{P}(u|f)$. Donc, pour un f fixé, qui maximise $\mathbf{P}(f|u) \mathbf{P}(u)$
- Lien avec méthodes variationnelles : revient à **minimiser**

$$E(u) = -\log \mathbf{P}(u) - \log \mathbf{P}(f|u)$$

- probabilité **a priori** $\mathbf{P}(u)$ ← **critère de régularité**

- en général  moins probable que  (si images "naturelles")




- débruitage : image régulière plus probable que du bruit




- segmentation :  moins probable que 

42 / 149

Méthodes markoviennes

- **vraisemblance $P(f|u)$** : modélise le processus de formation de l'image

- débruitage : si $u =$  alors  plus probable que 

- segmentation : si $u =$  alors  plus probable que 

- Hypothèse de base : $P(f|u)$, $P(u)$ plus facile à modéliser que $P(u|f)$ (alors que c'est ce dernier qui nous intéresse) \rightarrow utilisation de la formule de Bayes

43 / 149

Méthodes markoviennes en segmentation

- probabilité **a priori** $P(u)$

- $P(\text{noisy image}) = ?$ $P(\text{segmented image}) = ?$ $P(\text{butterfly mask}) = ?$

- critère de régularité pour des images à m classes \rightarrow modèle de Potts

$$P(u) = \frac{1}{Z} \exp \left(-\beta \sum_{\langle s,t \rangle} \varphi(u_s, u_t) \right)$$

où $\langle s, t \rangle =$ "les pixel s et t sont voisins" et $\varphi(a, b) = \begin{cases} -1, & \text{si } a = b \\ 1, & \text{si } a \neq b \end{cases}$

si $\beta > 0 \rightarrow P(\text{noisy image}) < P(\text{segmented image}) < P(\text{butterfly mask})$

44 / 149

Méthodes markoviennes en segmentation

Modélisation Bayésienne Markovienne :

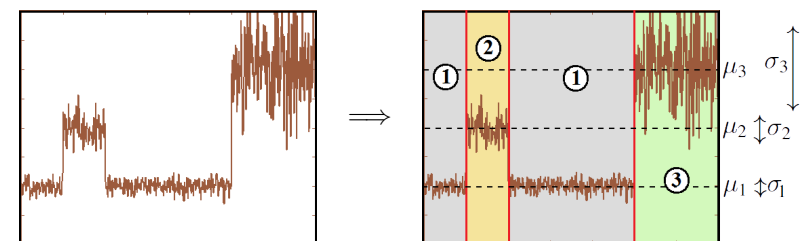
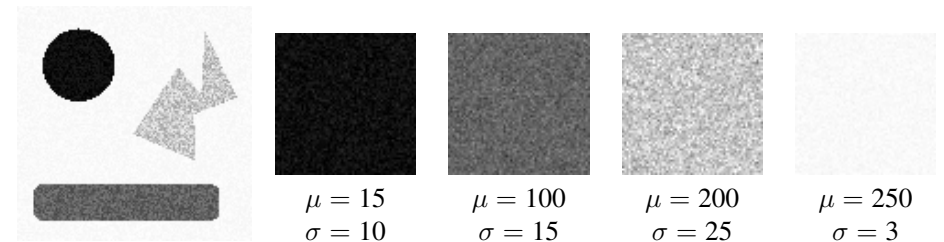
- **vraisemblance des observations $P(f|u)$**

$P(\text{noisy image} | \text{butterfly mask}) = ?$ $P(\text{boat image} | \text{butterfly mask}) = ?$ $P(\text{segmented butterfly image} | \text{butterfly mask}) = ?$

- on connaît u (répartition et localisation des différentes zones texturées)
- Pour un u donné, la probabilité $P(f|u)$ sera grande (par définition) si f est "cohérent" avec u , i.e. si les textures présentes dans f sont dans les zones indiquées par u
- on choisit de représenter une "texture" par : cste + bruit blanc
- Pour un u donné (à valeurs dans $\{0, \dots, M-1\}$), on choisit donc M textures, c'est à dire M moyennes μ_i et M écart-types σ_i
- On choisit donc de dire que la valeur de f au point s , qui fait partie de la classe $i = u_s$, va suivre une loi gaussienne : $\frac{1}{\sigma_i \sqrt{2\pi}} \exp \left(-\frac{(f_s - \mu_i)^2}{2\sigma_i^2} \right)$

45 / 149

Méthodes markoviennes en segmentation



46 / 149

Méthodes markoviennes en segmentation

- vraisemblance des **observations** (proba. de transition) :

- loi gaussienne :

$$F_s | U_s = i \sim \mathcal{N}(\mu_i, \sigma_i)$$

$$\mathbf{P}(F_s = f_s | U_s = i) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp\left(-\frac{(f_s - \mu_i)^2}{2\sigma_i^2}\right)$$

$$\mathbf{P}(F = f | U = u) = \prod_s \mathbf{P}(F_s = f_s | U_s = u_s)$$

- Remarque** : en quoi la probabilité $\mathbf{P}(F_s = f_s | U_s = i)$ dépend-elle de U ?
- L'étiquette donnée par u n'intervient donc "que" par i : si on sait que le pixel s est d'étiquette u_s alors la probabilité de f_s est donnée par une loi gaussienne de moyenne μ_{u_s} et d'écart-type σ_{u_s}

47 / 149

Méthodes markoviennes en segmentation

- Modélisation Bayésienne Markovienne :

- a posteriori** :

$$\mathbf{P}(U = u | F = f) \propto \mathbf{P}(F = f | U = u) \mathbf{P}(U = u)$$

$$\propto \left(\prod_s \frac{1}{\sigma_{u_s} \sqrt{2\pi}} \exp\left(-\frac{(f_s - \mu_{u_s})^2}{2\sigma_{u_s}^2}\right) \right) \exp\left(-\beta \sum_{\langle s,t \rangle} \phi(u_s, u_t)\right)$$

$$\propto \exp\left[\ln\left(\prod_s \frac{1}{\sigma_{u_s} \sqrt{2\pi}} \exp\left(-\frac{(f_s - \mu_{u_s})^2}{2\sigma_{u_s}^2}\right)\right) - \beta \sum_{\langle s,t \rangle} \phi(u_s, u_t)\right]$$

$$\propto \exp\left[\sum_s \ln\left(\frac{1}{\sigma_{u_s} \sqrt{2\pi}}\right) - \sum_s \frac{(f_s - \mu_{u_s})^2}{2\sigma_{u_s}^2} - \beta \sum_{\langle s,t \rangle} \phi(u_s, u_t)\right]$$

$$\propto \exp\left[-\sum_s \ln \sigma_{u_s} - \sum_s \frac{(f_s - \mu_{u_s})^2}{2\sigma_{u_s}^2} - \beta \sum_{\langle s,t \rangle} \phi(u_s, u_t)\right]$$

48 / 149

Méthodes markoviennes en segmentation

- On a donc construit un modèle probabiliste qui donne la probabilité d'avoir une segmentation u étant donné une image f (à un facteur de normalisation près)
- Cette probabilité peut s'écrire sous la forme d'un champ de Gibbs/Markov :

$$\mathbf{P}(U = u | F = f) \propto \exp\left(-H(u, f)\right)$$

$$\text{avec } H(u, f) = \sum_s \ln \sigma_{u_s} + \sum_s \frac{(f_s - \mu_{u_s})^2}{2\sigma_{u_s}^2} + \beta \sum_{\langle s,t \rangle} \phi(u_s, u_t)$$

- Remarque** : Maximiser $\mathbf{P}(U = u | F = f)$ revient à minimiser $H(u, f)$

49 / 149

Méthodes markoviennes en segmentation

Maximum A Posteriori

estimateur MAP (Maximum A Posteriori) :

- u tel que $\mathbf{P}(U = u | F = f)$ maximum (pour f donné)
- i.e. u tel que $H(u, f)$ minimum

- Comment minimiser ? énergie trop irrégulière pour descente de gradient ou similaire
- Algorithme stochastique → **recuit simulé**
- Besoin d'un outil permettant d'échantillonner une distribution de probabilité → **Algorithme de Metropolis-Hastings**

50 / 149

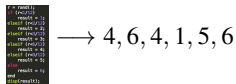
Méthodes markoviennes en segmentation

Échantillonnage : Introduction

- réaliser le tirage d'une config. en suivant une loi de probabilité donnée ?
- Exemples

- Tirage à Pile ou Face : $\mathbf{P}(Pile) = \mathbf{P}(Face) = \frac{1}{2}$
- Lancer d'un dé truqué : 3 fois plus de chance pour $\{4, 5, 6\}$ que pour $\{1, 2, 3\}$

$$\mathbf{P}(D = i) = p_i, p_1 = p_2 = p_3 = \frac{1}{12}, p_4 = p_5 = p_6 = \frac{3}{12}$$



→ 4, 6, 4, 1, 5, 6

- Pour champs de Markov/Gibbs : $\pi(x) = \frac{1}{Z} \exp\left(-\sum_{C \in \mathcal{C}} V_C(x_C)\right) \rightarrow ?$



Méthodes markoviennes en segmentation

Échantillonnage : Introduction

Problème

- Donnée : un champ de Markov π , i.e. un système de voisinage et des potentiels de cliques
- Comment tirer une réalisation selon la loi $\pi(X) \propto \exp\left(-\sum_{C \in \mathcal{C}} V_C(x_C)\right)$? On ne connaît généralement pas le facteur de normalisation Z .

Méthodes de Monte-Carlo

- Toute méthode d'approximation utilisant un générateur de nombres (pseudo-)aléatoires

Chaîne de Markov Monte-Carlo (MCMC)

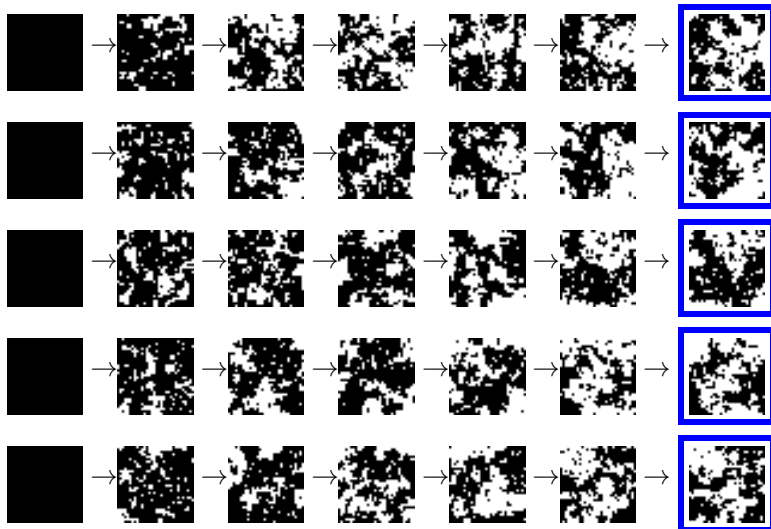
- but** : générer une chaîne de Markov X^0, X^1, \dots, X^m "convergeant" vers la distribution cible π

$$\forall x^0 \in \Omega, \mathbf{P}(X^m = x | X^0 = x^0) \xrightarrow{m \rightarrow +\infty} \pi(x)$$

- moyen** : concevoir des "remises à jours" exploitant la décomposition locale du champ de Markov

Méthodes markoviennes en segmentation

Échantillonnage : Introduction



Méthodes markoviennes en segmentation

Échantillonnage : Introduction



Méthodes markoviennes en segmentation

Algorithme de Metropolis

Principe

- On veut échantillonner la loi

$$\pi(x) = \frac{1}{Z} \exp(-H(x)) = \frac{1}{Z} \exp\left(-\sum_C V_C(x_C)\right)$$

- Principe : Transition pour un site courant s
 - Tirer une valeur λ uniformément dans Λ (ens. des valeurs possibles)
 - Proposer** λ comme nouvelle valeur pour le site s (définissant une nouvelle image y)
 - L'accepter** si elle fait diminuer H
 - L'accepter** avec probabilité $\exp(H(x) - H(y))$ si elle fait augmenter H
 - Si λ est rejetée alors x_s ne change pas

55 / 149

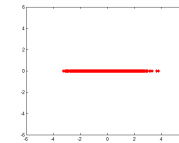
Méthodes markoviennes en segmentation

Algorithme de Metropolis

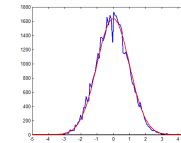
Exemple 1 : simulation de la loi normale $\mathcal{N}(0, 1)$ 1D

- Loi gaussienne de moyenne 0 et d'écart-type 1 : $f_X(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$
- On choisit un intervalle d'étude : $[-6, 6]$
- On tire aléatoirement une valeur initiale dans cet intervalle : x
- on tire une nouvelle valeur y qui sera acceptée avec une probabilité :

$$\text{Probabilité d'acceptation de } y : \min\left(1, \exp\left(\frac{x^2 - y^2}{2}\right)\right)$$



Itération : 1000



Répartition (50 000 tirages)

56 / 149

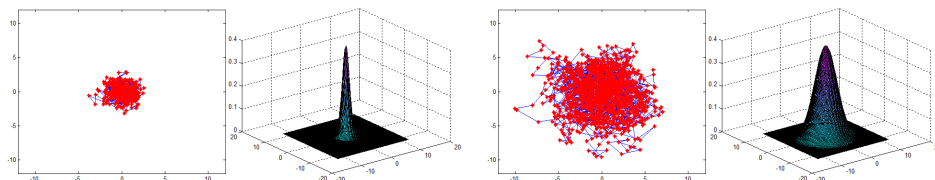
Méthodes markoviennes en segmentation

Algorithme de Metropolis

Exemple 2 : simulation de la loi normale $\mathcal{N}(0, \sigma)$ 2D

- Loi gaussienne de moyenne 0 et d'écart-type σ : $f_X(x, y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2+y^2}{2\sigma^2}}$

DEMO MATLAB



57 / 149

Méthodes markoviennes en segmentation

Algorithme de Metropolis

Algorithme final

- choisir une image initiale x^0 aléatoirement
- à l'étape m , remettre à jour le site $s_m[n]$ selon la procédure précédente
- au bout d'un certain temps les images x^{r^n} obtenues suivent la loi $\pi(x)$

Application aux modèles d'Ising et Potts ($B = 0$)

DEMO MATLAB



$\beta = 0$

$\beta = 0.5$

$\beta = 1$

$\beta = 10$

58 / 149

Méthodes markoviennes en segmentation

- Les méthodes d'échantillonnage vu précédemment produisent des configurations "**typiques**". Exemples :
 - Pièce truquée : proba $p > 0.5$ de faire pile. On effectue 1000 lancers, les échantillons (construits par Metropolis par exple) ont environ une proportion p de piles.
 - quel est le x tel que $\pi(x)$ soit maximal ? → "que des piles"
 - Pourquoi n'observera-t-on "jamais" cette configuration en échantillonnant ?
 $\mathbf{P}(\text{"600 piles puis 400 faces"}) < \mathbf{P}(\text{"que des piles"}) < \mathbf{P}(\text{"proportion } p \text{ de piles"})$
 - Exemple : deux lancers et $p = 0.6$

$$\mathbf{P}(PP) = 0.6 * 0.6 \quad \mathbf{P}(PF) = 0.6 * 0.4 \quad \mathbf{P}(FP) = 0.4 * 0.6 \quad \mathbf{P}(FF) = 0.4 * 0.4$$

$$\mathbf{P}(PP) = 0.36 \quad \mathbf{P}(PF) = 0.24 \quad \mathbf{P}(FP) = 0.24 \quad \mathbf{P}(FF) = 0.16$$

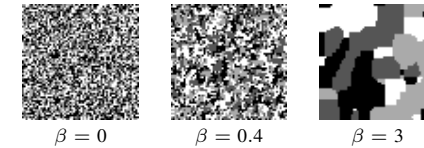
$$\mathbf{P}(PP) = 0.36 \quad \mathbf{P}(\text{"un pile et un face"}) = \mathbf{P}(PF) + \mathbf{P}(FP) = 0.48$$
 - 1000 lancers et $p = 0.6$: $\mathbf{P}(\text{"que des piles"}) = p^{1000} \approx 1.4 * 10^{-222}$
 $\mathbf{P}(\text{"600 piles puis 400 faces"}) = p^{600} (1-p)^{400} \approx 5.1 * 10^{-293}$
 $\mathbf{P}(\text{"600 piles et 400 faces (ordre indifférent)}) = \binom{1000}{600} p^{600} (1-p)^{400} \approx 0.025$

59 / 149

Méthodes markoviennes en segmentation

- Les méthodes d'échantillonnage vu précédemment produisent des configurations "**typiques**". Exemples :

- π , Ising/Potts (Metropolis)



- x les plus probables ? **images constantes**

- $\pi(\text{image noire}) = \pi(\text{image blanche}) > \pi(\text{image structurée}) \quad \forall \beta$

- Même "phénomène" que précédemment

$$\pi(\text{image blanche}) < \pi(\text{"une image qui ressemble à [image structurée]"})$$

60 / 149

Méthodes markoviennes en segmentation

Recuit simulé

- Le recuit simulé est une méthode inspirée d'un processus utilisé en métallurgie (et datant de la préhistoire !). Ce processus alterne des cycles de refroidissement lent et de réchauffage (recuit) qui tendent à minimiser l'énergie du matériau.
- Le recuit simulé s'appuie sur l'algorithme de Metropolis-Hastings (dans notre cas, l'algorithme de Metropolis ou l'échantillonneur de Gibbs) → **BUT : trouver le Maximum A Posteriori, la configuration la plus probable.**
- Étant donné une distribution de Gibbs $\pi(x) = \frac{1}{Z} \exp(-H(x))$, on cherche le x tel que $H(x)$ soit minimal
- Considérons la distribution de Gibbs **avec température** associée :

$$\pi_T(x) = \frac{1}{Z_T} \exp\left(-\frac{H(x)}{T}\right)$$

$$\text{avec } H(x) = \sum_{C \in \mathcal{C}} V_C(x_C) \quad \text{et} \quad Z_T = \sum_{y \in \Omega} \exp\left(-\frac{H(y)}{T}\right)$$

61 / 149

Méthodes markoviennes en segmentation

Recuit simulé : Comportement aux températures limites

- Comment se comporte $\frac{\pi_T(y)}{\pi_T(x)}$ lorsque $T \rightarrow 0$ et $T \rightarrow \infty$?

- $\frac{\pi_T(y)}{\pi_T(x)} = \exp\left(-\frac{H(y) - H(x)}{T}\right)$

- $T \rightarrow \infty \Rightarrow \frac{\pi_T(y)}{\pi_T(x)} \rightarrow 1$

- $T \rightarrow 0 \Rightarrow \frac{\pi_T(y)}{\pi_T(x)} \rightarrow 0$, si $H(y) > H(x)$

62 / 149

Méthodes markoviennes en segmentation

Recuit simulé : Démonstration pour $T \rightarrow +\infty$

$$\pi_T(x) = \frac{\exp\left(-\frac{H(x)}{T}\right)}{\sum_{y \in \Omega} \exp\left(-\frac{H(y)}{T}\right)}$$

$$\pi_T(x) = \frac{1}{\sum_{y \in \Omega} \exp\left(-\frac{H(y) - H(x)}{T}\right)}$$

$$\pi_T(x) \xrightarrow{T \rightarrow +\infty} \frac{1}{\text{card } \Omega}, \quad \forall x \in \Omega$$

Plus la température est élevée et plus la distribution π_T se rapproche de la distribution uniforme sur Ω .

63 / 149

Méthodes markoviennes en segmentation

Recuit simulé : Démonstration pour $T \rightarrow 0$

- On pose $H^* = \min_{x \in \Omega} H(x)$ et $\Omega^* = \arg \min_{x \in \Omega} H(x) = \underbrace{\{x \in \Omega \mid H(x) = H^*\}}_{\text{ce qu'on cherche, } x^{\text{map}}}$

- On multiplie, dans $\pi_T(x)$, le numérateur et le dénominateur par $\exp\left(\frac{H^*}{T}\right)$

$$\pi_T(x) = \frac{\exp\left(-\frac{H(x) - H^*}{T}\right)}{\sum_{y \in \Omega} \exp\left(-\frac{H(y) - H^*}{T}\right)} = \frac{\exp\left(-\frac{H(x) - H^*}{T}\right)}{\text{card } \Omega^* + \sum_{y \in \Omega, y \notin \Omega^*} \exp\left(-\frac{H(y) - H^*}{T}\right)}$$

- et donc finalement : $\pi_T(x) \xrightarrow{T \rightarrow 0} \begin{cases} 0 & \text{si } x \notin \Omega^* \\ \frac{1}{\text{card } \Omega^*} & \text{si } x \in \Omega^* \end{cases}$

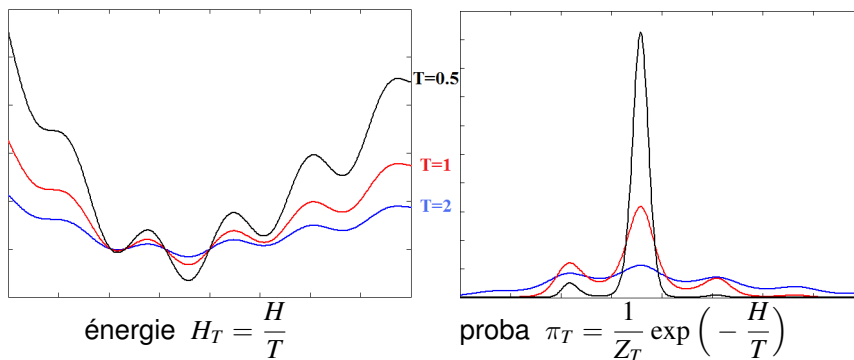
Plus la température est faible et plus la distribution π_T se rapproche de la distribution uniforme sur Ω^* et est nulle ailleurs.

64 / 149

Méthodes markoviennes en segmentation

Recuit simulé : Effet de la température

Quand $T \rightarrow 0$, si on échantillonne π_T on a de plus en plus de chances de "tomber" sur x^{map}



DEMO MATLAB

65 / 149

Méthodes markoviennes en segmentation

Recuit simulé

- Si on échantillonne le champs de Gibbs avec température (par Metropolis-Hastings) et que l'on fait diminuer la température à chaque itération, va-t-on minimiser H ?

Théorème (Geman and Geman 1984)

Il existe une température initiale T_0 telle que

$$\text{si } T_n \geq \frac{T_0}{\ln(1+n)} \text{ (pour tout } n) \text{ alors } \lim_{n \rightarrow +\infty} \mathbf{P}(X^n \in \Omega^*) = 1, \quad \forall X^0.$$

- Si T_n diminue très lentement (logarithmiquement), alors l'algorithme de Metropolis ou l'échantillonneur de Gibbs (en prenant T_n à l'itération n) converge vers un **minimum global** de l'énergie.
- Condition théorique : $T_0 = \max_{s, \lambda, \mu} |H(x^{s, \lambda}) - H(x^{s, \mu})|$

66 / 149

Méthodes markoviennes en segmentation

Recuit simulé

- Lien avec physique statistique : silicate fondu refroidi trop rapidement → matériaux métastable (verre) au lieu de l'état fondamental d'énergie minimale (cristal)
- Théorème → résultat théorique très intéressant

le recuit simulé peut trouver le minimum global de toute énergie, si on le laisse chercher indéfiniment.

- En pratique : "indéfiniment" = "trop long" !
- On utilise donc des heuristiques de refroidissement inspirées de la méthode initiale mais qui vont converger plus rapidement ... mais pas forcément vers le minimum global
- exemples :
 - $T_n = T_0 a^n$ avec $a \approx 0.98$ ou refroidissement par palier : $T_n = T_0 a^{\lfloor n/p \rfloor}$
 - critère d'arrêt (sur le nb de transitions "nulles")

67 / 149

Méthodes markoviennes en segmentation

Recuit simulé

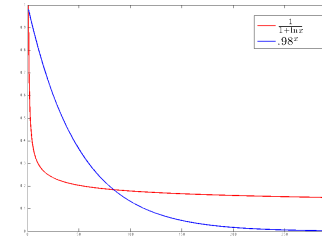
Comparaison entre

- La décroissance **théorique** du théorème (qui assure la convergence)

$$T_n \geq \frac{T_0}{\ln(1+n)}$$

- Celle utilisée classiquement **en pratique** (qui assure un temps de calcul plus "raisonnable")

$$T_n = T_0 a^n \quad \text{avec} \quad a \approx 0.98 \text{ par exemple}$$



68 / 149

Méthodes markoviennes en segmentation

méthode ICM (Iterated Conditional Modes)

- Sorte de recuit simulé à température nulle (proposé par Besag en 1983)
- algorithme déterministe : au site courant **maximiser la descente d'énergie**

- ▷ Initialisation x^0 proche de la solution
- ▷ Suite d'images x^n
- ▷ à l'étape n , on balaye tous les sites s et

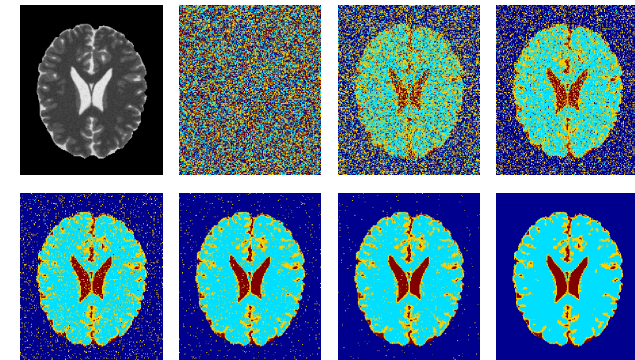
$$x_s^n \leftarrow \arg \max_{\lambda \in \Lambda} \mathbf{P}(X_s = \lambda | y, x_{\partial(s)}^{n-1})$$

Propriétés

- Algorithme **déterministe**, résultat dépendant de l'initialisation
- Convergence rapide (quelques balayages)
- Risque de converger vers un minimum local de H

69 / 149

Exemple : segmentation texturale (recuit simulé)



■ matière blanche ■ matière grise ■ liquide céphalo-rachidien

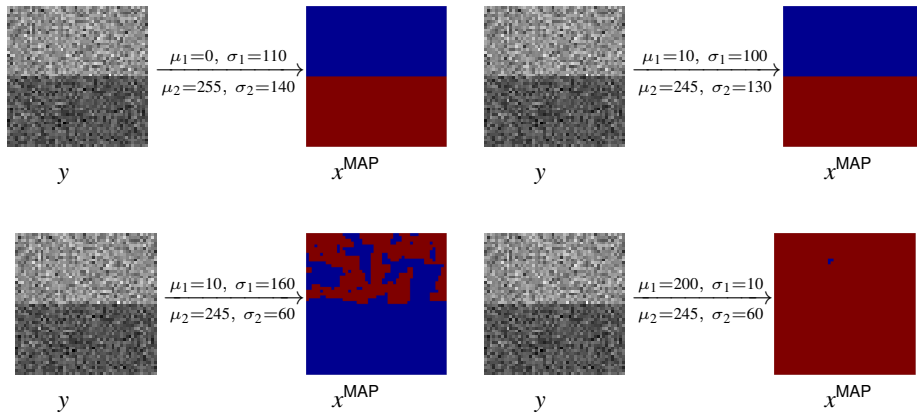
DEMO MATLAB

DEMO MATLAB

70 / 149

Méthodes markoviennes en segmentation

- Problème : dépend beaucoup des choix qu'on a fait pour μ_i et σ_i

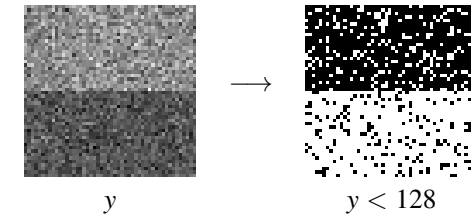


- Comment estimer/choisir ces paramètres *avant* de lancer l'optimisation ?

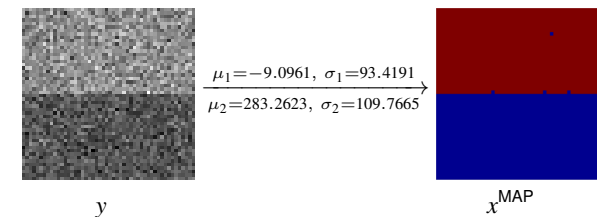
71 / 149

Méthodes markoviennes en segmentation

- utiliser une première segmentation basique → exemple : seuillage



- $\text{mean}(Y(Y < 128)) \rightarrow -9.0961$ $\text{mean}(Y(Y \geq 128)) \rightarrow 283.2623$
- $\text{std}(Y(Y < 128)) \rightarrow 93.4191$ $\text{std}(Y(Y \geq 128)) \rightarrow 109.7665$



72 / 149

Méthodes markoviennes en segmentation

- Mieux (mais plus long) → utiliser une méthode d'espérance-maximisation

● Algorithme EM¹

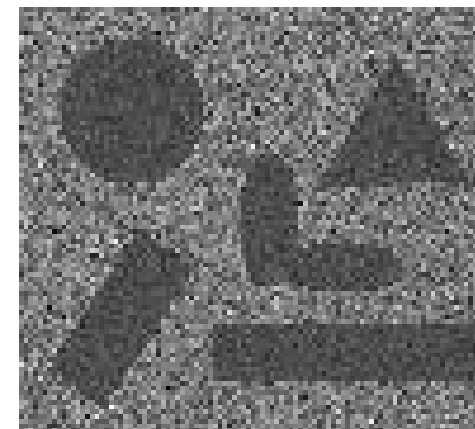
- On initialise les paramètres μ_i et σ_i , puis on alterne entre
 - 1 Segmentation en utilisant les μ_i et σ_i actuels
 - 2 Remise à jour des μ_i et σ_i , en calculant moyenne et écart-type de chaque classe obtenue à l'étape précédente
- On s'arrête lorsque les paramètres ne changent plus suffisamment (et donc la segmentation associée non plus)

1. http://fr.wikipedia.org/wiki/Algorithme_espérance-maximisation

73 / 149

Méthodes markoviennes en segmentation

Algorithme EM. exemple : segmentation basée texture



$$\begin{array}{ll} \mu_1 = 128 & \sigma_1 = 30 \\ \mu_2 = 255 & \sigma_2 = 90 \end{array}$$

DEMO MATLAB

74 / 149

Méthodes markoviennes en segmentation

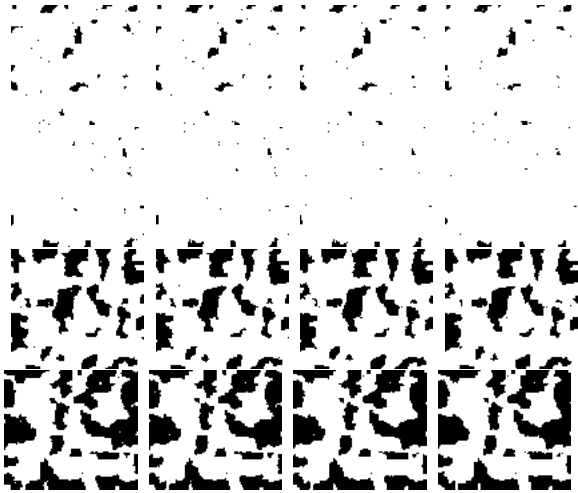
Algorithme EM. exemple : segmentation basée texture

$$\begin{aligned} \mu_1 &= 50 & \sigma_1 &= 155 \\ \mu_2 &= 100 & \sigma_2 &= 60 \end{aligned}$$

$$\begin{aligned} \mu_1 &= 283 & \sigma_1 &= 260 \\ \mu_2 &= 212 & \sigma_2 &= 111 \end{aligned}$$

$$\begin{aligned} \mu_1 &= 275 & \sigma_1 &= 112 \\ \mu_2 &= 215 & \sigma_2 &= 96 \end{aligned}$$

$$\begin{aligned} \mu_1 &= 264 & \sigma_1 &= 94 \\ \mu_2 &= 197 & \sigma_2 &= 93 \end{aligned}$$



75 / 149

Méthodes markoviennes en segmentation

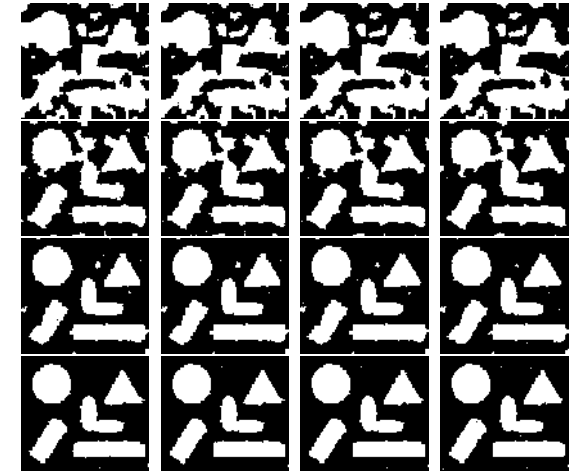
Algorithme EM. exemple : segmentation basée texture

$$\begin{aligned} \mu_1 &= 261 & \sigma_1 &= 90 \\ \mu_2 &= 181 & \sigma_2 &= 89 \end{aligned}$$

$$\begin{aligned} \mu_1 &= 262 & \sigma_1 &= 89 \\ \mu_2 &= 167 & \sigma_2 &= 81 \end{aligned}$$

$$\begin{aligned} \mu_1 &= 259 & \sigma_1 &= 89 \\ \mu_2 &= 147 & \sigma_2 &= 65 \end{aligned}$$

$$\begin{aligned} \mu_1 &= 257 & \sigma_1 &= 90 \\ \mu_2 &= 130 & \sigma_2 &= 39 \end{aligned}$$



$$\begin{aligned} \mu_1 &= 256 & \sigma_1 &= 90 \\ \mu_2 &= 127 & \sigma_2 &= 30 \end{aligned}$$

76 / 149

Méthodes markoviennes

Bilan

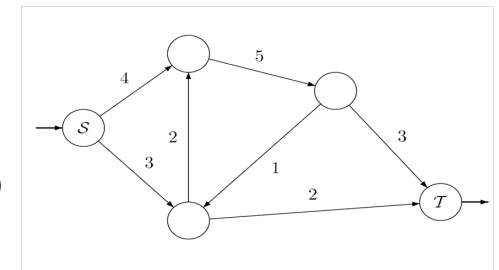
- par rapport aux approches continues/variationnelles : modélisation du bruit, ajout d'a priori
- Très à la vogue dans les années 85-95 : elles ont été moins utilisées lorsque les images sont devenues trop grandes (temps de calcul)
- Regain d'intérêt depuis 5/10 ans grâce à l'apparition des techniques de coupure de graphe (Graph Cut) qui permettent une optimisation plus rapide

77 / 149

Approches d'optimisation par graphe : Méthode des "Graph-Cuts"²

Notations

- $G = (V, E)$: Graphe Orienté
- V : sommets, E : arêtes
- pour chaque $(v_1, v_2) \in E$, une capacité w_{v_1, v_2} (poids des arêtes)
- S : source, T : puits

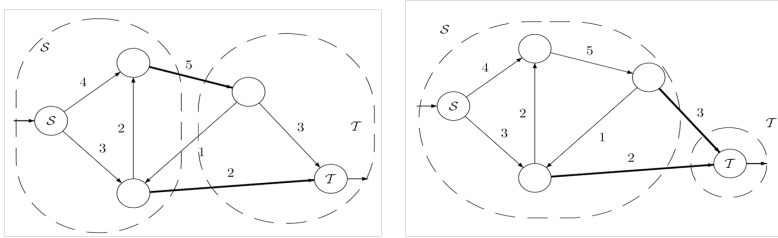


2. adapté de <http://mickaelpetchaud.free.fr/graphcuts.pdf>

78 / 149

Approches d'optimisation par graphe : Méthode des "Graph-Cuts"

Coupe



- coupe = partition du graphe en deux régions S et T ($S \in S$ et $T \in T$)
- poids d'une coupe =
$$\sum_{\substack{(v_1, v_2) \in E \\ v_1 \in S, v_2 \in T}} w_{v_1, v_2}$$
- coupe minimale = coupe de poids minimal

79 / 149

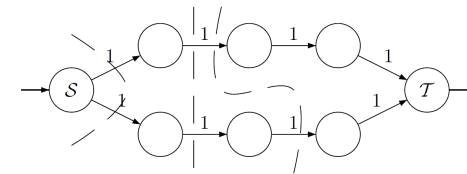
Approches d'optimisation par graphe : Méthode des "Graph-Cuts"

Théorème

Rechercher une coupe minimale dans un graphe est un problème P.

Remarques :

- beaucoup de problèmes sur les graphes sont NP (exemple : recherche de la coupe maximale)
- il peut y avoir plusieurs solutions



80 / 149

Approches d'optimisation par graphe : Méthode des "Graph-Cuts"

Reformulation en terme de flot maximal

On appelle flot, une fonction $f : E \mapsto \mathbb{R}$ vérifiant

- pour tout sommet $p \in V$ autre que S ou T , on a :

$$\sum_{e=(p, \cdot) \in E} f(e) = 0 \quad (1)$$

- pour toute arête $e \in E$, on a

$$f(e) \leq w_e \quad (2)$$

- (1) \rightarrow conservation du flot en chaque sommet
- (2) \rightarrow une arête ne peut contenir un flot dépassant sa capacité
- on définit la "valeur du flot"

$$\sum_{e=(S, \cdot) \in E} f(e) = \sum_{e=(\cdot, T) \in E} f(e)$$

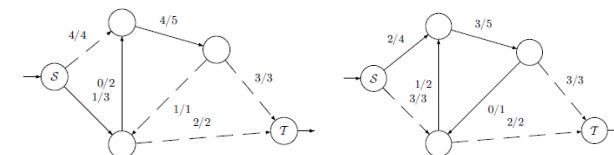
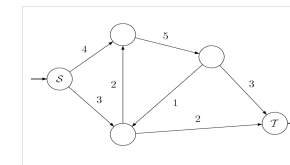
81 / 149

Approches d'optimisation par graphe : Méthode des "Graph-Cuts"

Reformulation en terme de flot maximal

Théorème

Pour un graphe G vérifiant nos hypothèses, la valeur d'une coupe minimale est égale à la valeur d'un flot maximal.



82 / 149

Approches d'optimisation par graphe : Méthode des "Graph-Cuts"

- De plus toute arête intervenant dans une coupe minimale est une arête saturée par un flot maximal
- Si on a trouvé un flot maximal, on regarde les arêtes saturées et on supprime itérativement les arêtes inutiles
- Pour trouver flot maximal, plusieurs méthodes :
 - **saturation de chemins** : à partir du flot nul, trouver itérativement un chemin de S à T sur lequel il n'y a pas d'arête saturée. On rajoute alors autant de flot que possible à ce chemin
 - **poussage de flot** : on envoie autant de flot que l'on peut à partir de S . On se retrouve alors avec des noeuds dits actifs, c'est à dire qui reçoivent un excès de flot. On pousse alors ce flot excessif vers d'autres noeuds disponibles.

83 / 149

Méthode des "Graph-Cuts" : utilisation en image

- Les "graph-cut" vont nous servir pour minimiser une énergie de la forme :

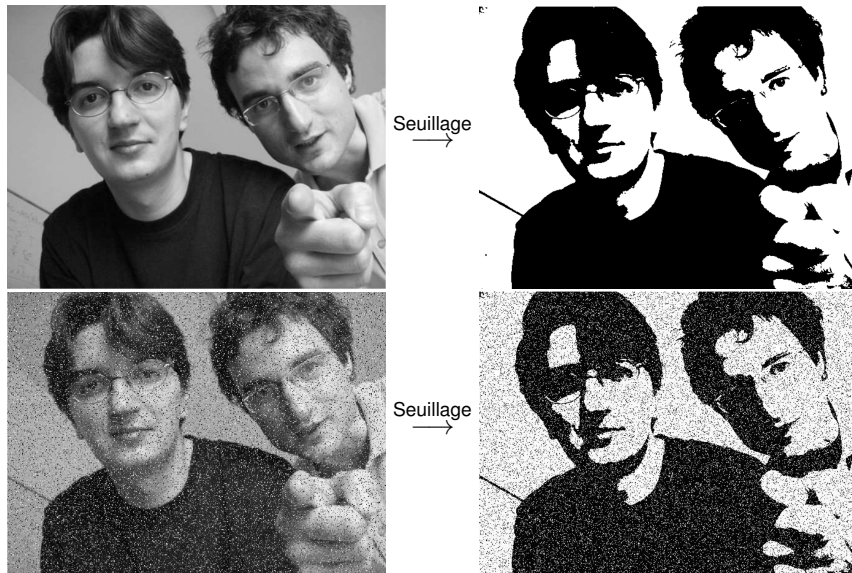
$$E(x) = \sum_i D_i(x_i) + \alpha \sum_{i,j} V_{ij}(x_i, x_j)$$

- typiquement :
 - $D \rightarrow$ attache aux données :
 - $V \rightarrow$ critère de régularité (non nul uniquement si i et j sont voisins)
- Exemple : segmentation binaire d'une image bruitée I ($0 \leq I_i \leq 1$)
 - pour chaque pixel i , on cherche donc une étiquette $x_i \in \{0, 1\}$
 - on prend

$$\begin{cases} D_i(0) = I_i \\ D_i(1) = 1 - I_i \end{cases} \quad \text{et} \quad V_{ij}(x_i, x_j) = \begin{cases} 0 & \text{si } x_i = x_j \\ 1 & \text{sinon} \end{cases}$$

84 / 149

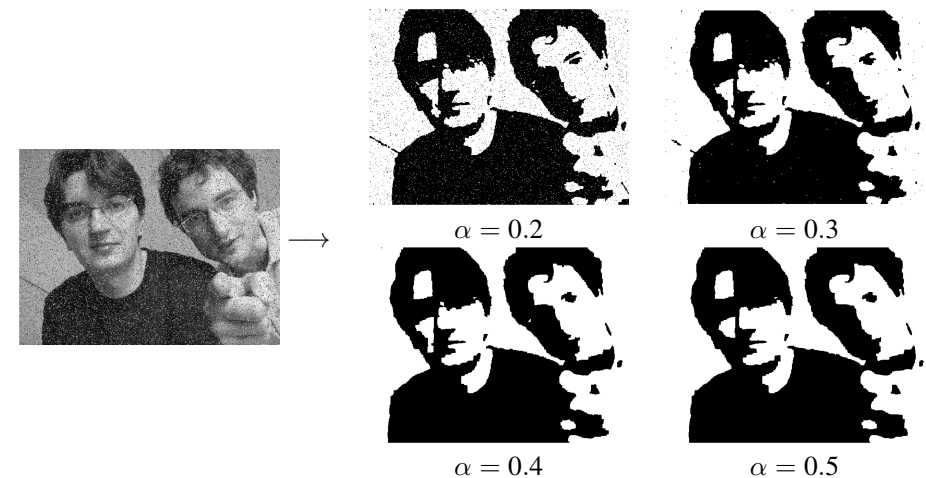
Méthode des "Graph-Cuts" : utilisation en image



85 / 149

Méthode des "Graph-Cuts" : utilisation en image

Minimisation (par Graph-Cuts) de l'énergie précédente



86 / 149

Méthode des "Graph-Cuts" : utilisation en image

- Les "graph-cut" vont nous servir pour minimiser une énergie de la forme :

$$E(x) = \underbrace{\sum_i D_i(x_i)}_{\text{terme de fidélité, attache aux données}} + \alpha \underbrace{\sum_{i,j} V_{ij}(x_i, x_j)}_{\text{régularisation}}$$

- Lien avec le cadre Bayésien :

$$\mathbf{P}(X|I) = \mathbf{P}(I|X) \mathbf{P}(X)$$

$$\Rightarrow \log(\mathbf{P}(X|I)) = \underbrace{\log(\mathbf{P}(I|X))}_{\text{terme de fidélité, attache aux données}} + \underbrace{\log(\mathbf{P}(X))}_{\text{régularisation}}$$

- Dans le cas général, minimiser

$$E(x) = \sum_i D_i(x_i) + \alpha \sum_{i,j} V_{ij}(x_i, x_j)$$

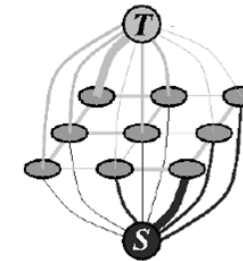
est un problème NP-complet ("arg")

87 / 149

Méthode des "Graph-Cuts" : utilisation en image

Principe général

- On transforme le problème de minimisation en problème de coupe minimal dans un graphe
- On construit un graphe constitué d'un sommet par variable de l'énergie (pour chaque pixel)
- On ajoute une source et un puits, qui vont représenter en quelque sorte les valeurs possibles de chaque variable (0 pour la source et 1 pour le puits)

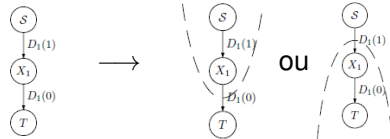


88 / 149

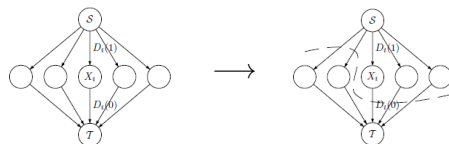
Méthode des "Graph-Cuts" : utilisation en image

exemples simples

- Un seul pixel : on veut minimiser $D_1(x_1)$.



- Si $D_1(0) < D_1(1)$ la solution est donc $x_1 = 0$ et réciproque.
- avec $D_1(0) = I_1$, $D_1(1) = 1 - I_1 \rightarrow$ seuillage à $\frac{1}{2}$
- n pixels sans interaction ($\alpha = 0$) : on veut minimiser $\sum_i D_i(x_i)$.

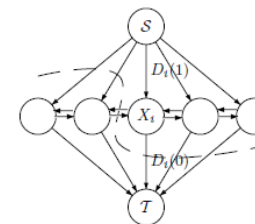


Indep. des variables donc solution : $x_i = 0$ si $D_i(0) < D_i(1)$ (seuillage à $\frac{1}{2}$)

89 / 149

Méthode des "Graph-Cuts" : utilisation en image

- n pixels avec interactions ($\alpha > 0$)



- il faut que la valeur d'une coupe, corresponde à l'énergie de la configuration associée Ainsi :
Coupe minimale \leftrightarrow énergie minimale
- Les poids des arêtes sont plus compliqués à définir (en fonction de l'énergie, i.e. de D et V)
- Principe : deux voisins seront reliés par une arête de poids fort s'ils sont de valeurs proches

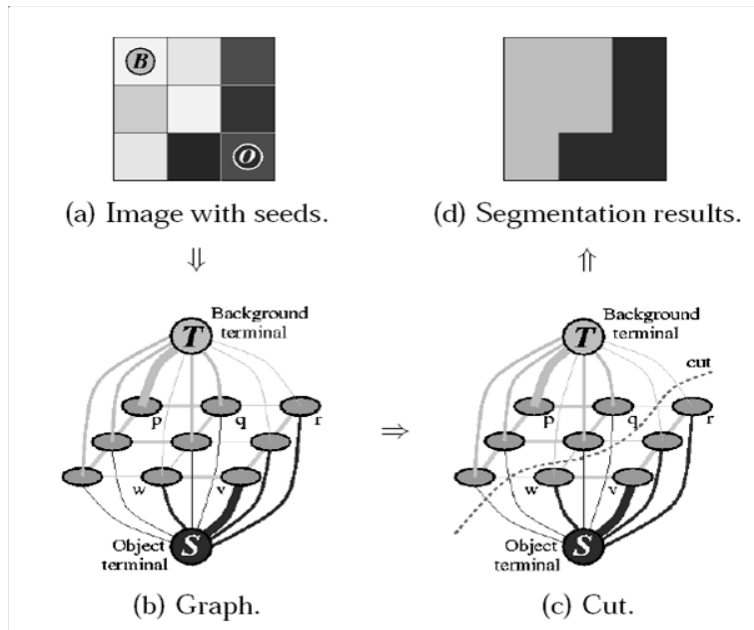
Attention !

Toutes les énergies de la forme précédente ne sont pas minimisables par Graph-Cut

- Mais si l'énergie à minimiser vérifie certaines propriétés, alors on trouve le **minimum global** et la minimisation est **rapide**

90 / 149

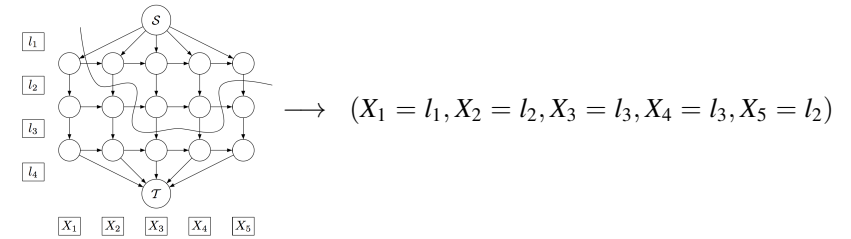
Méthode des "Graph-Cuts" :segmentation



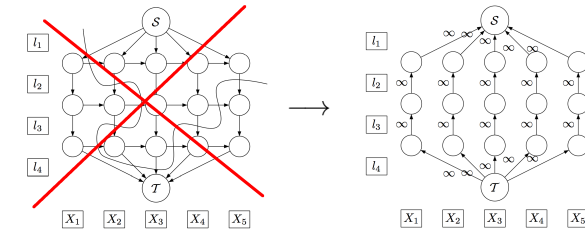
91 / 149

Méthode des "Graph-Cuts" :segmentation

- Le cas multi-étiquette



- Attention, des précautions à prendre :



92 / 149

Méthode des "Graph-Cuts" :segmentation

- Le cas multi-étiquette



4 niveaux de gris

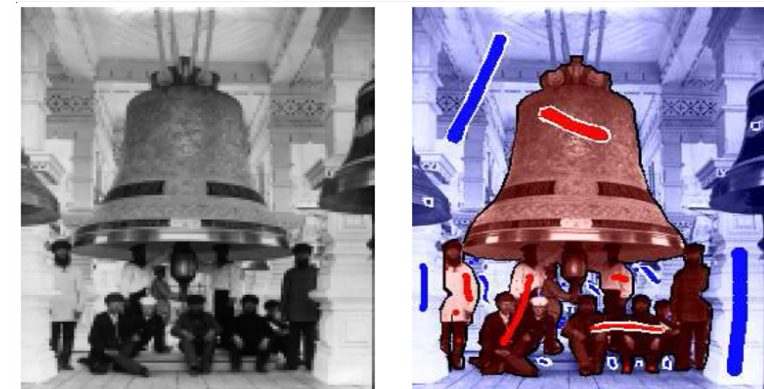


16 n.d.g

64 n.v.g

93 / 149

Méthode des "Graph-Cuts" : segmentation



Boykov and Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. 2001. On peut initialiser avec des "graines" (liens infini entre le puit et l'objet et la source et le fond par exemple)

94 / 149

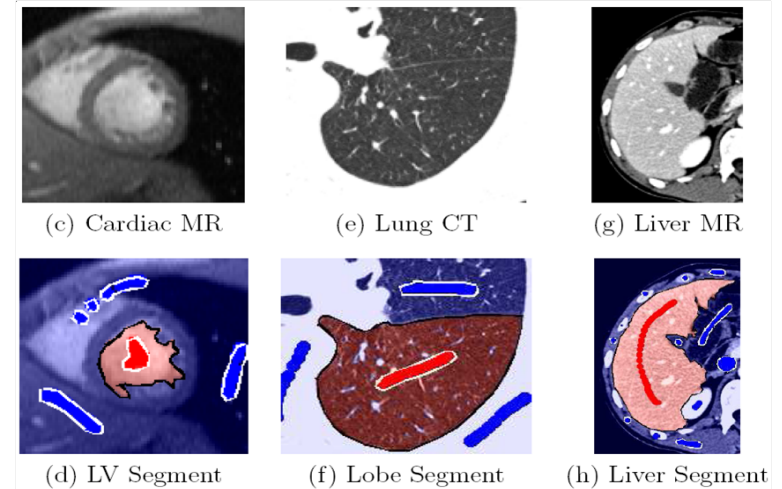
Méthode des "Graph-Cuts" : segmentation



Juan.O. and Keriven.R. Trimap segmentation for fast and user-friendly alpha matting. 2005

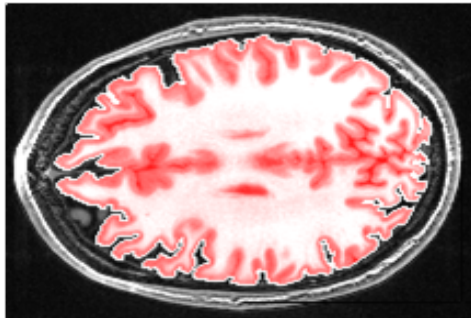
95 / 149

Méthode des "Graph-Cuts" : segmentation



96 / 149

Méthode des "Graph-Cuts" : segmentation



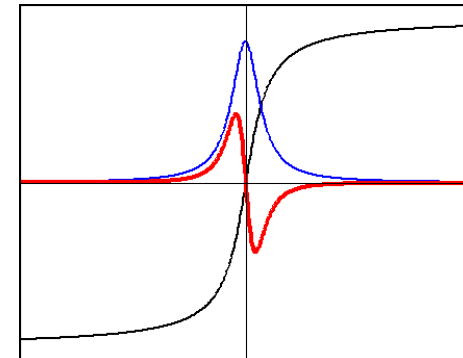
- Temps moyen en 2D (438×522) : 250ms
- Temps moyen en 3D ($256 \times 256 \times 124$) : 9.5s
- Pour aller plus loin

<http://mickaelpetchaud.free.fr/graphcuts.pdf>
et références associées

97 / 149

Approches "contours"

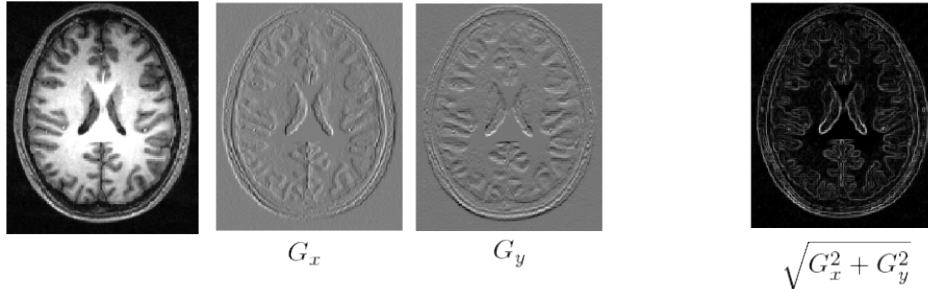
- contour = ensemble des pixels séparant deux régions
- correspond à une importante variation locale d'intensité
 - on regarde la variation d'**intensité** autour d'un contour
 - la **dérivée première** passe par un maximum
 - la **dérivée seconde** passe par un zéro



99 / 149

Gradient de l'image

$$G = \overrightarrow{\text{grad}} f(x, y) \quad \begin{cases} G_x = \frac{\partial f}{\partial x}(x, y) \\ G_y = \frac{\partial f}{\partial y}(x, y) \end{cases}$$



100/149

Seuillage de la norme du gradient de l'image lissée

$$\sigma = 2$$



seuil = 5

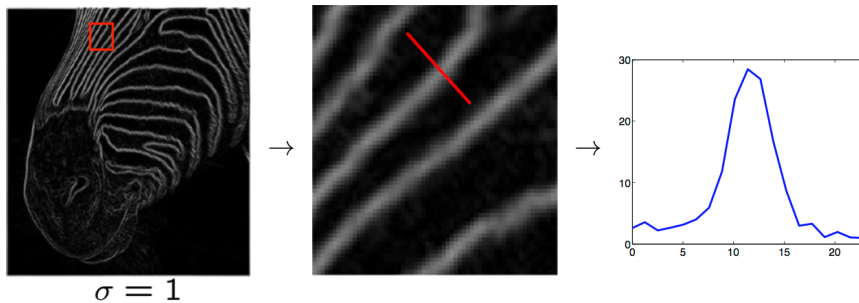
seuil = 10

seuil = 15

101/149

Détecteur de Canny

- Profil dans la direction du gradient



- Principe : ne garder que les maxima locaux (dans la direction du gradient) et supérieurs à un seuil

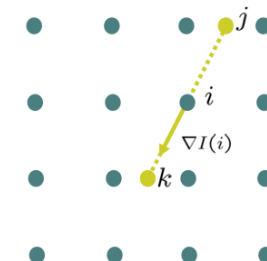
102/149

Détecteur de Canny

- Principe : ne garder que les maxima locaux (dans la direction du gradient) et supérieurs à un seuil

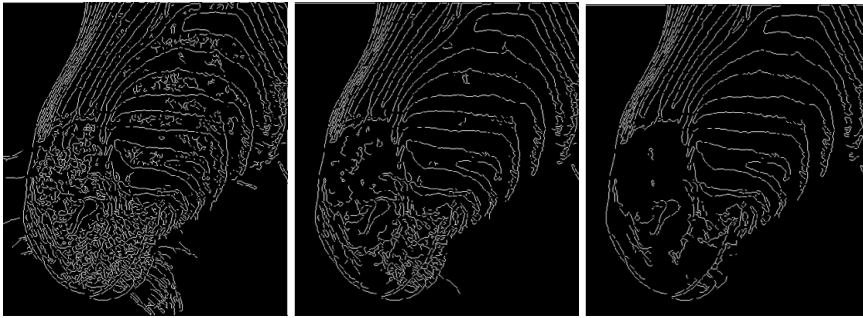
contour en i si

- $|\nabla I(i)| > |\nabla I(j)|$
- $|\nabla I(i)| > |\nabla I(k)|$
- $|\nabla I(i)| > \text{seuil}$



103/149

Détecteur de Canny, résultats



104/149

Détecteur de Canny, résultats



105/149

Détecteur de Canny, résultats



106/149

Détecteur de Canny, résultats

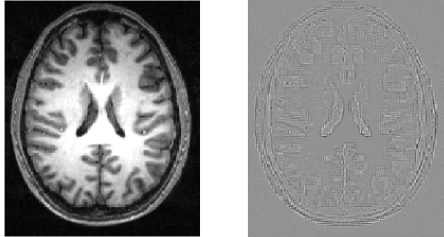


107/149

Laplacien de l'image

$$\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

"équivalent de la dérivée seconde" en 2D

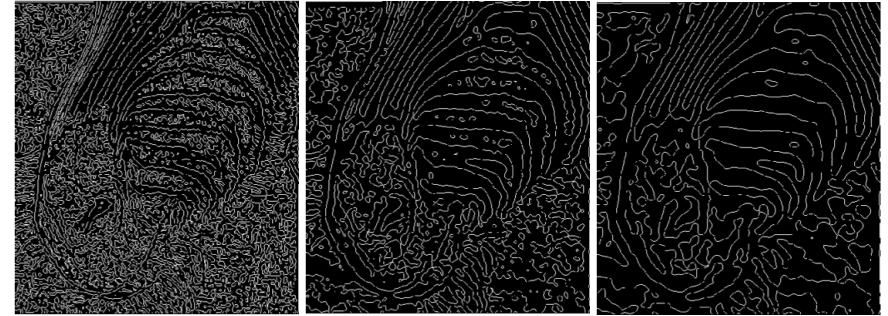


Détection des passages par 0 du laplacien (Marr et Hildreth, 1976).

108 / 149

Passages par zéros du laplacien

- Lignes de niveau zéro (courbes fermées) du laplacien lissé (LoG) et interpolé spatialement



- Mais, soit trop de lignes (lissage faible), soit peu de lignes mais en partie décalées (lissage fort)

109 / 149

Approches "contours"

Seuillage par hystérésis

- compromis dans le choix du seuil sur la norme du gradient pour sélectionner les contours
 - fermeture des contours (peu de faux négatif)
 - immunité au bruit (peu de faux positifs)
- Choix de deux seuils, un haut (spécificité) et un bas (sensibilité)
 - 1 On sélectionne les points au-dessus du seuil haut
 - 2 pour tout point entre le seuil haut et le seuil bas, on regarde s'il existe un point de son voisinage > seuil haut
- Principe utilisé dans le détecteur de Canny.

110 / 149

Approches "contours"

Seuillage par hystérésis



Seuillage simple
S=50

Seuillage simple
S=100

Seuillage par
hystérésis
Sh=100, Sb=50

111 / 149

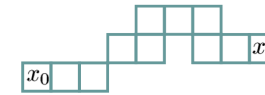
Extraction de chaînes de contours

- **Problème** : extraction d'une chaîne de contours entre deux points donnés
- **Applications** :
 - analyse automatique de certains types d'images (extraction de routes dans images satellite, d'artères en angiographie, etc.)
 - découpe interactive en édition d'image
- **Approche classique**
 - définir une fonction de coût pour une chaîne candidate
 - trouver la chaîne de coût minimum entre les deux extrémités par programmation dynamique : problème combinatoire de plus court chemin dans un graphe résolu par l'algorithme de Dijkstra

112/149

Coût d'une chaînes de pixels

- Chaîne : succession de pixels voisins au sens de la 4 ou 8 connexité



- Minimiser une somme de coûts élémentaires positifs

$$E_n(x_0, \dots, x_n; I) = \sum_{i=1}^n \phi(x_{i-1}, x_i; I)$$

- Exemple avec f décroissante et g croissante

$$\phi(x_{i-1}, x_i; I) = \alpha \|x_i - x_{i-1}\| + \beta f(\|\nabla I(x_i)\|) + \gamma g(|\nabla I(x_i)| \cdot (x_i - x_{i-1}))$$

113/149

Lignes de partage des eaux (watershed)

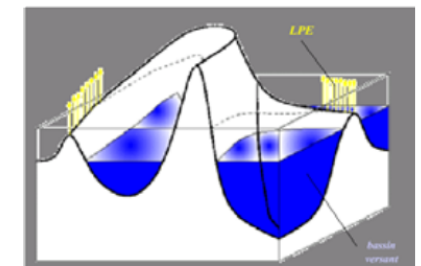
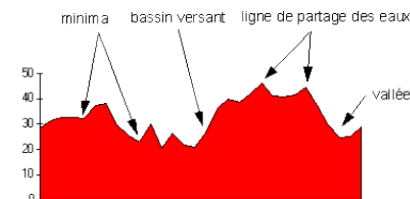
- Digabel et Lantuéjoux, 1972
- l'image est représenté par un relief que l'on inonde progressivement
- pour la segmentation, on travaille sur l'image de la norme du gradient



114/149

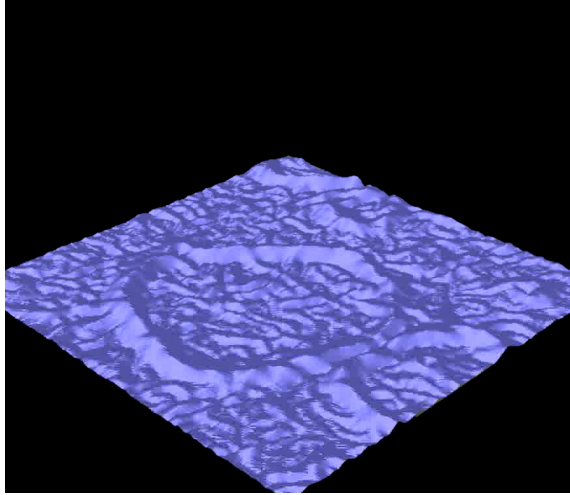
Lignes de partage des eaux (watershed)

- Points de départ : minima locaux
- montée des eaux
- barrage élémentaire à chaque rencontre de bassins : ligne de partage des eaux



115/149

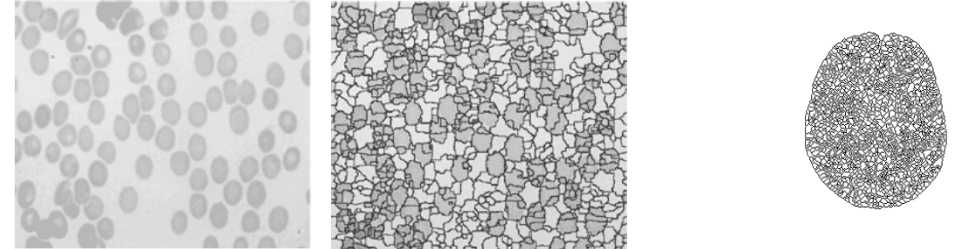
Lignes de partage des eaux (watershed)



<http://pmc.polytechnique.fr/~vta/water.mpeg>

116/149

Lignes de partage des eaux (watershed)



- algorithme non local
- problème de sur-segmentation
- sensibilité à tout minimum local, au bruit (→ lissage préventif)

117/149

Contours actifs

- Idée générale : on cherche la forme (courbe/surface) \mathcal{C} qui minimise une énergie

$$E(\mathcal{C}) = E_{\text{image}}(\mathcal{C}) + E_{\text{interne}}(\mathcal{C})$$

- $E_{\text{image}}(\mathcal{C})$ va essayer d'attirer la courbe sur les contours de l'image
- $E_{\text{interne}}(\mathcal{C})$ permet d'ajouter des a priori sur la forme de la courbe que l'on cherche (régularité)
- Contours actifs géodésiques :

$$E(\mathcal{C}) = \int_0^1 |\mathcal{C}'(p)|^2 dp + \beta \int_0^1 g_{\mathcal{I}}(\mathcal{C}(p)) dp$$

- la fonction de potentiel $g_{\mathcal{I}}$ vise à attirer la courbe vers les contours de l'objet à segmenter. Elle dépend donc de **l'image \mathcal{I} à segmenter**.
- Minimisation par "descente de gradient" : Euler-Lagrange

118/149

Approches "contours"

Contours actifs / Level-set



119/149

Contours actifs : exemple de fonction de potentiel

- pour la segmentation on veut avoir : $g_{\mathcal{I}}(x, y)$ proche de 1 loin des contours et minimal à proximité d'un contour
- ainsi, $\int_0^1 g_{\mathcal{I}}(\mathcal{C}(p)) dp$ sera petit uniquement si \mathcal{C} coïncide avec un contour.
- exemple :

$$g_{\mathcal{I}}(x, y) = \frac{1}{1 + |\nabla \mathcal{I}(x, y)|}$$

- pas robuste au bruit \rightarrow on lisse avant de prendre le gradient :

$$g_{\mathcal{I}}(x, y) = \frac{1}{1 + |\nabla (G_{\sigma} * \mathcal{I}(x, y))|}$$

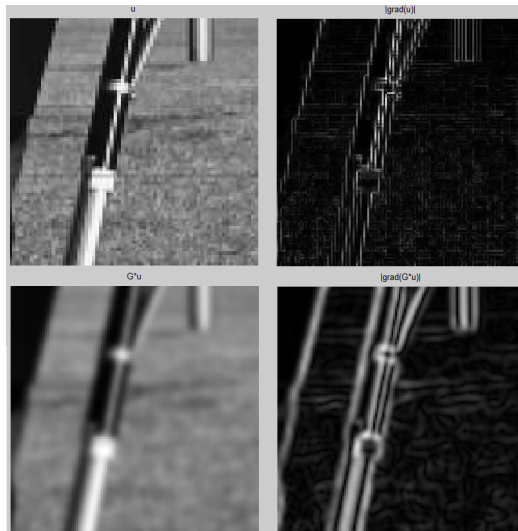
120 / 149

fonction de potentiel image



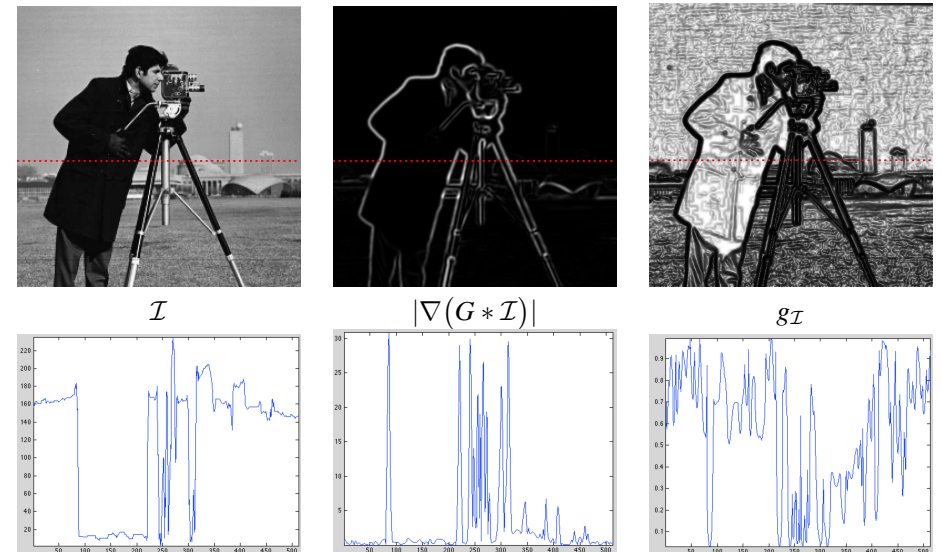
121 / 149

fonction de potentiel image



122 / 149

Contours actifs ou snakes : fonction de potentiel image



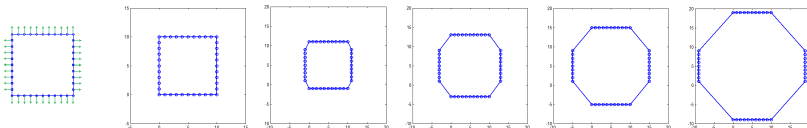
123 / 149

Contours actifs ou snakes : difficultés

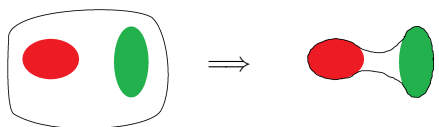
- Dépend de la paramétrisation de la courbe

$$\begin{cases} x(p) = r \sin(p 2 \pi) \\ y(p) = r \cos(p 2 \pi) \end{cases} \Rightarrow \text{Image d'un cercle} \Leftarrow \begin{cases} x(p) = r \cos(p 2 \pi) \\ y(p) = r \sin(p 2 \pi) \end{cases}$$

- problèmes de discrétisation de la courbe (liste de points)



- pas de changement de topologie possible



- en dimension supérieure (3D → évolution de surface) : complexité de l'implémentation, discrétisation, évolution.

124 / 149

Représentation par courbes de niveaux : méthode "Level Set"

- Représentation/caractérisation d'une courbe
 - paramétrisation
 - liste de points
 - implicite ← **Level Set**

Level Set : Osher-Sethian, 1988.

Idée principale :

- la courbe \mathcal{C} est représentée de manière **implicite**.
- \mathcal{C} est représenté par une surface de dimension supérieure Φ dont elle est le **niveau zéro**

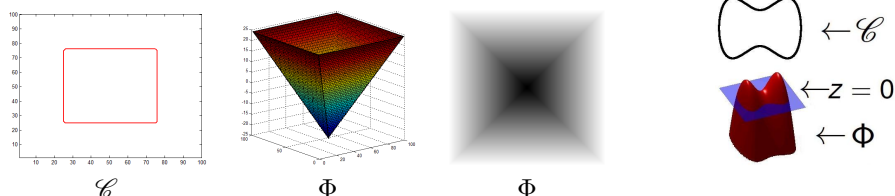
$$\mathcal{C} = \{ (x, y) \mid \Phi(x, y) = 0 \}$$

- Pour une courbe (dans \mathbb{R}^2) → ϕ est une image plane
- Pour une surface (dans \mathbb{R}^3) → ϕ est une image volumique

125 / 149

"Level Set"

- une courbe plane (dans \mathbb{R}^2) : représentée par une fonction $\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}$ (image, surface)

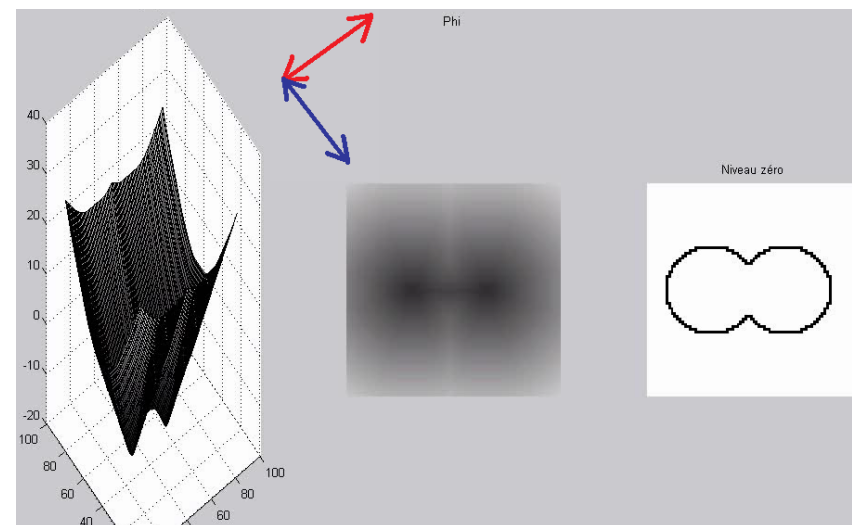


- un bon choix : Φ fonction distance signée à la courbe \mathcal{C} (négative à l'intérieur, positive à l'extérieur).
- Exple : distance signée à un cercle de rayon r et de centre $(0, 0)$?

$$\phi(x, y) = \sqrt{x^2 + y^2} - r$$

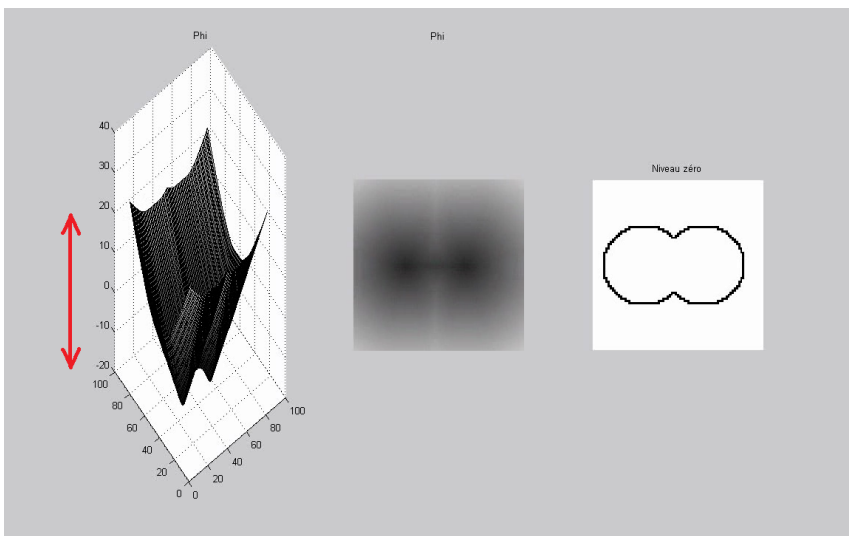
126 / 149

"Level Set" : lien évolution de \mathcal{C} / évolution de Φ



127 / 149

"Level Set" : lien évolution de \mathcal{C} / évolution de Φ



128 / 149

"Level Set" : lien évolution de \mathcal{C} / évolution de Φ

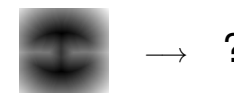
- on veut faire évoluer une courbe par des équations de la forme :

$$(1) \begin{cases} \frac{\partial \mathcal{C}}{\partial t}(t, p) = v(p) \mathbf{n}(p), \\ \mathcal{C}(0, p) = \mathcal{C}_0(p) \end{cases} \quad \text{exemple : } \begin{array}{c} \leftarrow \leftarrow \leftarrow \\ \leftarrow \leftarrow \leftarrow \\ \rightarrow \rightarrow \rightarrow \\ \rightarrow \rightarrow \rightarrow \end{array}$$

- la courbe $\mathcal{C}(t, p)$ bouge le long de ses normales avec une vitesse v
- un déplacement le long des tangentes ne change pas la forme de la courbe ("fait glisser" la courbe sur elle-même)
- représentation discrète (liste de points) :

$$\underbrace{\mathcal{C}((n+1)\Delta t, p_k)}_{\text{nouvelle position du point } n} \leftarrow \underbrace{\mathcal{C}(n\Delta t, p_k)}_{\text{ancienne position}} + \Delta t \underbrace{v(p_k) \mathbf{n}(p_k)}_{\text{vitesse}}$$

- représentation implicite : si \mathcal{C} vérifie (1), comment se comporte Φ ?



129 / 149

"Level Set" : lien évolution de \mathcal{C} / évolution de Φ

équation d'évolution sur \mathcal{C}

$$\begin{cases} \frac{\partial \mathcal{C}}{\partial t}(t, p) = v(p) \mathbf{n}(p), \\ \mathcal{C}(0, p) = \mathcal{C}_0(p) \end{cases} \quad (3)$$



équation d'évolution sur Φ

$$\begin{cases} \frac{\partial \Phi}{\partial t} = v |\nabla \Phi| \\ \Phi(0, x, y) = \Phi_0(x, y) \end{cases} \quad (4)$$

- condition initiale : $\Phi_0(x, y)$ tel que $\mathcal{C}_0(p) = \{(x, y) | \Phi_0(x, y) = 0\}$. exple : fonction distance signée \mathcal{C}_0
- Donc on fait évoluer Φ selon (4) et on extrait le niveau zéro **uniquement** pour afficher la courbe

130 / 149

Exemple : mouvement par courbure moyenne

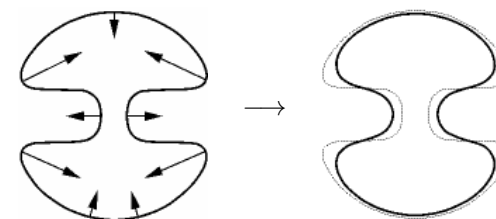
équation d'évolution sur \mathcal{C}

$$\begin{cases} \frac{\partial \mathcal{C}}{\partial t}(t, p) = \kappa(p) \mathbf{n}(p), \\ \mathcal{C}(0, p) = \mathcal{C}_0(p) \end{cases}$$



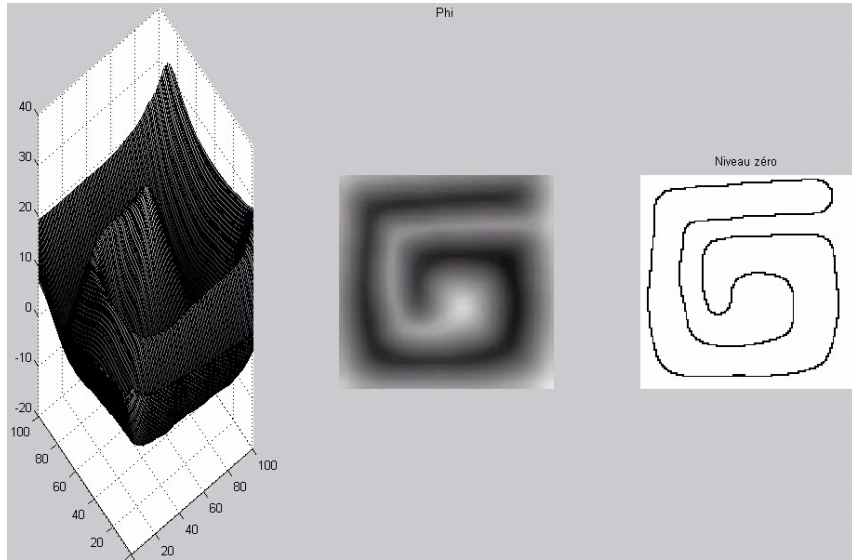
équation d'évolution sur Φ

$$\begin{cases} \frac{\partial \Phi}{\partial t} = \text{div} \left(\frac{\nabla \Phi}{|\nabla \Phi|} \right) |\nabla \Phi| \\ \Phi(0, x, y) = \Phi_0(x, y) \end{cases}$$



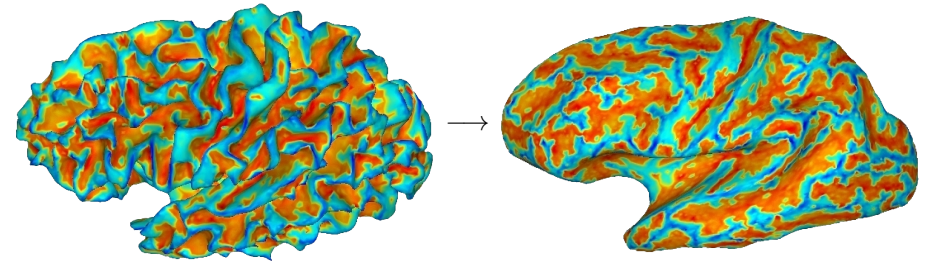
131 / 149

Exemple : mouvement par courbure moyenne



132/149

Exemple : mouvement par courbure moyenne



133/149

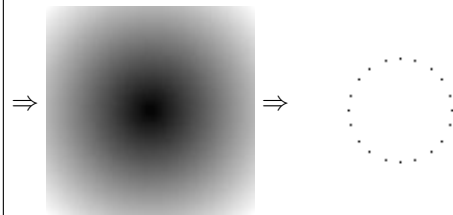
Extraction du niveau zéro

- On n'a besoin d'extraire le niveau zéro de Φ **uniquement** pour afficher la courbe correspondante. Mais comment faire ?
- extraire les pixels auxquels la fonction Φ vaut zéro \rightarrow bonne solution ?

exemple : fonction distance signée à un cercle

$$\phi(x, y) = \sqrt{x^2 + y^2} - r$$

qu'on représente sur une image $n \times n$



- sélectionner $|\Phi| < \epsilon$?

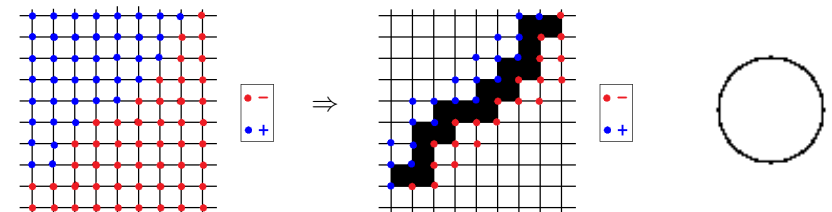
$\epsilon = 0.01$ $\epsilon = 0.03$ $\epsilon = 0.05$ $\epsilon = 0.1$ $\epsilon = 0.2$



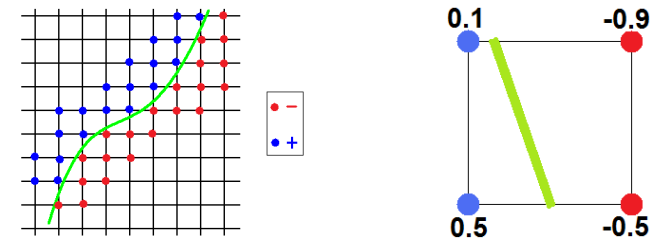
134/149

Extraction du niveau zéro

- détecter les changements de signe de Φ

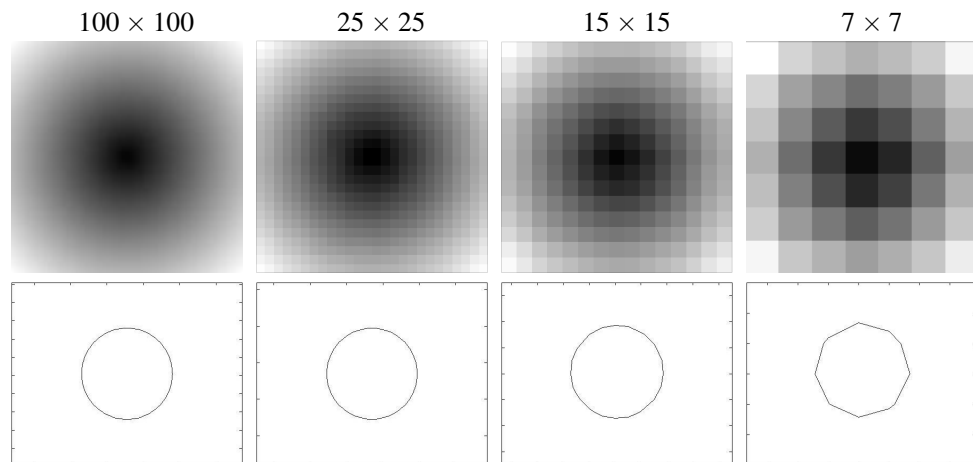


- mieux : interpolation en fonction des valeurs de Φ (fonction distance signée) sur ces pixels



135/149

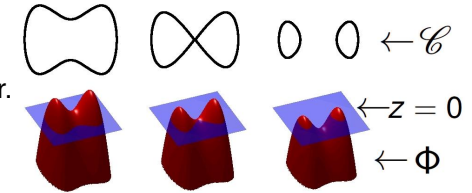
Extraction du niveau zéro



136/149

Intérêt de la représentation implicite

- Φ reste une fonction mais son niveau zéro (et donc \mathcal{C}) peut changer de topologie, se scinder, fusionner.



- **Pas besoin de prendre en compte numériquement ces changements topologiques**

- on utilise une **grille discrète fixée** → schéma aux différences finies
- des éléments géométriques **intrinsèque** à la courbe \mathcal{C} sont facilement exprimable à partir de Φ : normale $\mathbf{n} = -\frac{\nabla\Phi}{|\nabla\Phi|}$, courbure $\kappa = \text{div}\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right)$
- exple : cercle de rayon r et représenté par sa fctn dist signée, normale, courbure ?

$$\mathbf{n}(x, y) = \frac{-1}{\sqrt{x^2 + y^2}} \begin{pmatrix} x \\ y \end{pmatrix} \quad \kappa(x, y) = \frac{1}{\sqrt{x^2 + y^2}}$$

- même formulation (par Level Set) pour n'importe quelle **dimension** (courbe, surface) : implémentation aussi simple

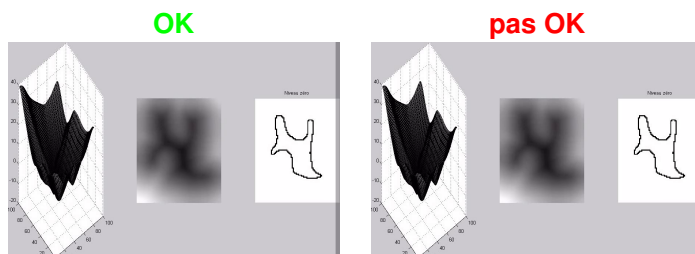
137/149

Représentation implicite : discrétisation

- Attention, de la même manière qu'en 1D, problèmes de stabilité possibles.
- Équation de transport $\frac{\partial v}{\partial t} + c \frac{\partial v}{\partial x} = 0$: si $c > 0$, schéma arrière. si $c < 0$, schéma avant

DEMO MATLAB

- cas d'un flot constant $v(t, p) = C$, propagation du front. Nécessité d'une discrétisation décentrée pour le calcul du gradient.



138/149

Retour à la segmentation

- évolution "Level Set" :

$$\frac{\partial \Phi}{\partial t} = \tilde{g}_{\mathcal{I}} \cdot |\nabla \Phi| \quad \text{où } \tilde{g}_{\mathcal{I}} \text{ l'extension de la fonction de potentiel } g_{\mathcal{I}}$$

- en général : données pas lisses, évolution chaotique (même si on a lissé avant de calculer $g_{\mathcal{I}}$), introduction de la courbure :

$$\frac{\partial \Phi}{\partial t} = \kappa \cdot \tilde{g}_{\mathcal{I}} \cdot |\nabla \Phi|$$

- κ : terme de lissage, ne dépend pas des données $\kappa = \text{div}\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right)$
- $\tilde{g}_{\mathcal{I}}$: facteur d'arrêt, dépendant des données

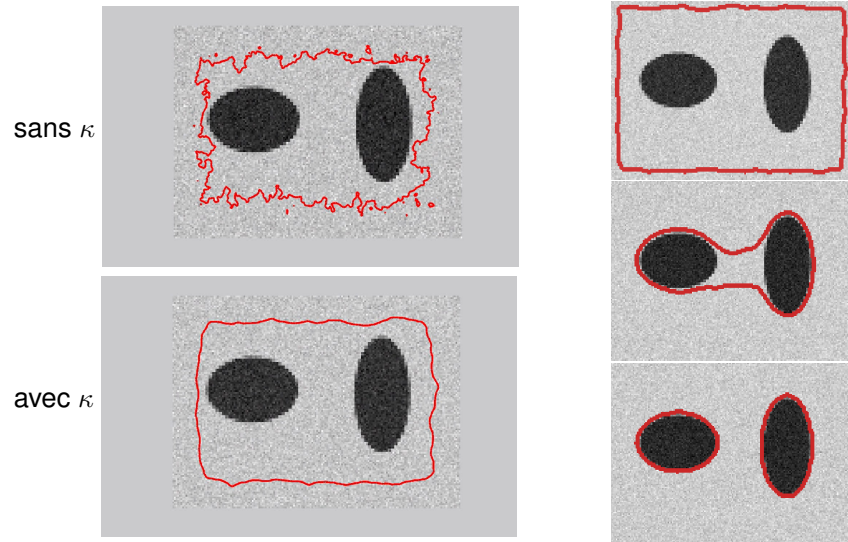
- si on connaît le sens global dans lequel doit se faire l'évolution :

$$\frac{\partial \Phi}{\partial t} = (\kappa + C) \cdot \tilde{g}_{\mathcal{I}} \cdot |\nabla \Phi|$$

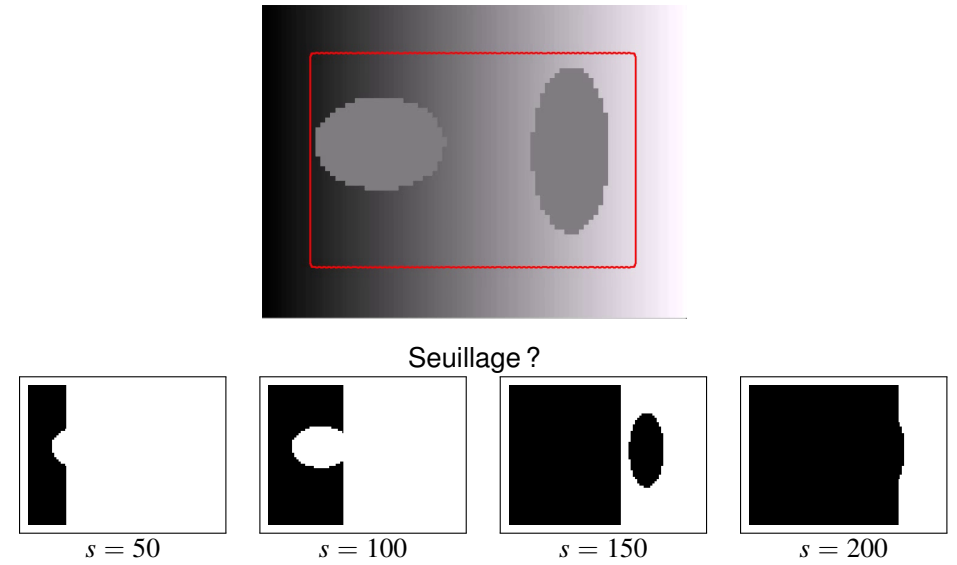
où C est une constante positive (contraction) ou négative (expansion)

139/149

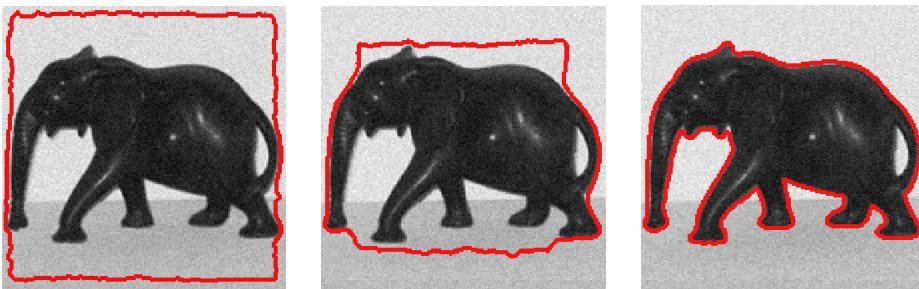
Segmentation par contours actifs : exemples



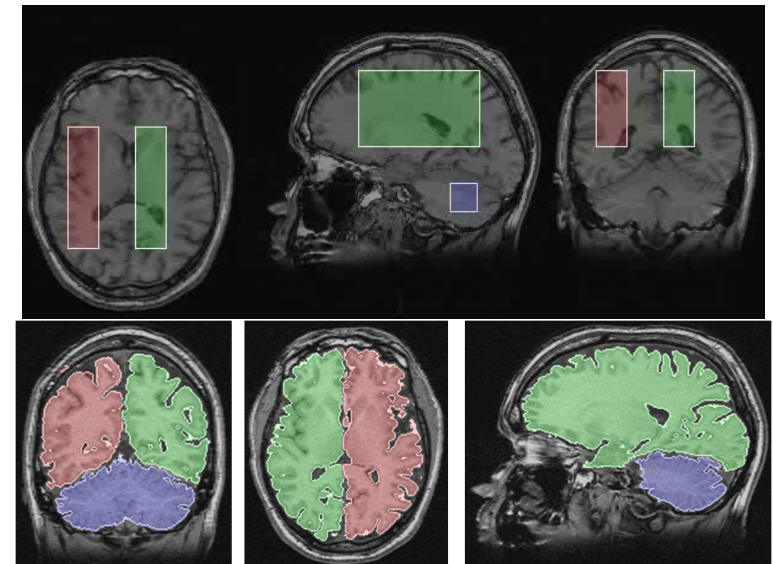
Segmentation par contours actifs : exemples



Segmentation par contours actifs : exemples



Segmentation par contours actifs : exemples



Représentation implicite : récapitulatif

Représentation Implicite

On représente une courbe \mathcal{C} , de manière implicite, par une surface de dimension supérieure Φ dont elle est le **niveau zéro**

$$\mathcal{C} = \{ (x, y) \mid \Phi(x, y) = 0 \}$$

- une courbe est représentée par une image (2D) $\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}$
- une surface est représentée par une image volumique (3D) $\Phi : \mathbb{R}^3 \rightarrow \mathbb{R}$

EDP sur $\mathcal{C} \leftrightarrow$ EDP sur Φ

$$\begin{cases} \frac{\partial \mathcal{C}}{\partial t}(t, p) = v(p) \mathbf{n}(p), \\ \mathcal{C}(0, p) = \mathcal{C}_0(p) \end{cases} \Leftrightarrow \begin{cases} \frac{\partial \Phi}{\partial t} = v |\nabla \Phi| \\ \Phi(0, x, y) = \Phi_0(x, y) \end{cases}$$

144 / 149

Représentation implicite : récapitulatif

Avantages :

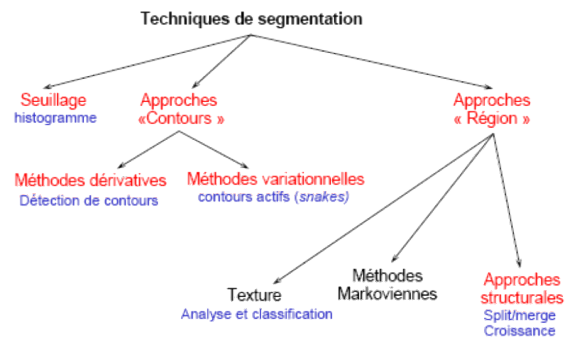
- changements de topologie "gratuits"
- discrétisation classique (grille discrète fixée \rightarrow schéma aux différences finies)
- des éléments géométriques **intrinsèques** à la courbe \mathcal{C} sont facilement exprimable à partir de Φ
- même formulation (par Level Set) pour n'importe quelle **dimension** (courbe, surface) : implémentation aussi simple

Inconvénients :

- temps de calculs
- impossible de représenter des courbes, ou surfaces, qui s'auto-intersectent
- difficile de représenter des courbes, ou surfaces, ouvertes

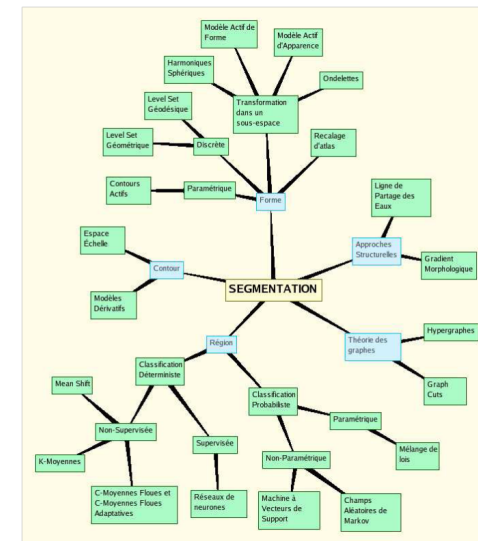
145 / 149

Récapitulatif



146 / 149

Récapitulatif (bis)



147 / 149

Imagerie médicale : segmentation basées sur un atlas

Atlas anatomique

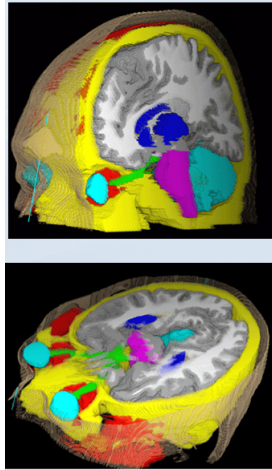
- image d'une anatomie moyenne
- segmentation associée effectuée par un expert

Objectif

- Utilisation de l'atlas pour segmenter une nouvelle image (patient)

Avantage

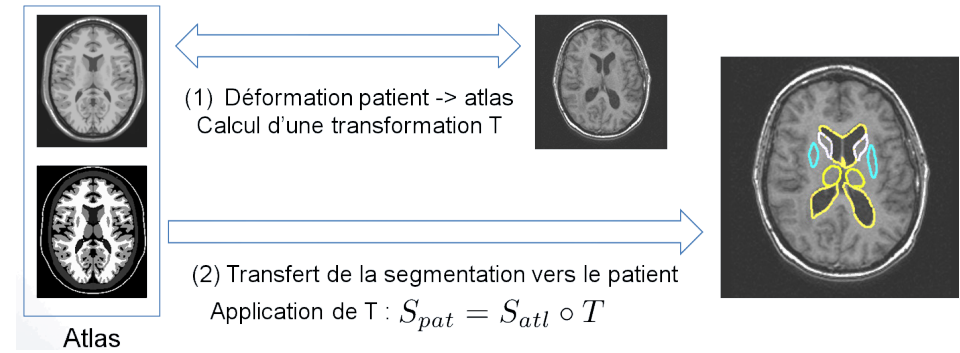
- prise en compte d'a priori spécifiques au résultat recherché
- Segmentation de multiples structures en une fois
- possibilité d'estimer des structures peu visible



148 / 149

Segmentation par Atlas : Méthode

- On transforme le problème de segmentation en un problème de **recalage**



On a donc besoin d'être capable de recalibrer une image sur une autre.

149 / 149